LEAK DETECTOR SYSTEM FOR POSITIVE AIRWAY PRESSURE MACHINES

by

AISHWARYA GOPALAKRISHNAN

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfilment

Of the Requirements

For the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

AUGUST 2016

Acknowledgements

First of all, I would like to thank the University of Texas at Arlington for letting me be a part of this University. I would like to express my deepest gratitude to my Thesis Supervisor, Professor Dr. Kambiz Alavi, *Professor and Associate chair of Electrical Engineering Department* at UTA, whom I have known since I started my Master's program and the one who constantly supported me and had faith in me throughout. He was very understanding and his encouragement was the driving force for me to give my best for the project.

I am extremely thankful to my supervisor Dr. Eileen Clements, *Director of Research* at University of Texas at Arlington Research Institute for giving me the opportunity to do this project and also for fulfilling my dream of being a part of UTARI by offering me the Graduate Research Assistant position. She has the attitude and the substance of a genius. I would also like to thank her for her patience and continuous support. She organised the project and guided me with it. Without her guidance, this thesis wouldn't have been possible.

I would also like to thank Dr. Khosrow Behbehani, *Dean of the University College of Engineering,* UTA for accepting to be a member of my Thesis Committee. I am also thankful to Manoj Mittal, Research Scientist at UTARI for helping me with his expertise and for being an excellent solution provider. I would also like to thank Castro Jose De Jesus, who works along with me at UTARI, for helping me with the hardware section of the project. I am also thankful to TexasMRC for funding the research.

I want to thank my friends Srividya Sekar, Ashok Nagamuthu Muniyandi and Gopinath Dayalan who have been a Family to me and who have been advising and cheering me for all the steps I take. Finally I want to thank my parents Mr. M.Gopalakrishnan and Mrs. M.S.Shantha for trusting, supporting and encouraging me throughout my life.

August 4, 2016

Abstract

LEAK DETECTOR SYSTEM FOR POSITIVE AIRWAY PRESSURE MACHINES

Aishwarya Gopalakrishnan

University of Texas at Arlington, 2015

Supervising Professor: Dr. Kambiz Alavi

The project aims to develop a system to indicate leaks occurring in a Positive Airway Pressure (PAP) machine which sends a passage of air to the patients suffering from sleep apnea, through a hose and a nose mask. The system uses programming software to record and analyse the data obtained from the sensors and provides an interface for the user to select a method for notifying the time instant and location of leak occurrences.

The system used an approach of using sensors for leak detection. Two pressure sensors and two audio sensors were selected out of which, one pressure sensor and audio sensor was installed on the mask side and the other pressure sensor and audio sensor was installed on the machine side. To install the sensors a connector was used for each side. The sensors were placed inside the connector. After the installation of the sensors, the signals from the sensors are acquired using a Data Acquisition device to read the analog signals and send it to the computer. A programming software was used to acquire the signals transmitted from the data acquisition device and perform necessary operations to analyse the data and detect leaks. The output of the sensors was too low for connecting with a data acquisition device, therefore circuits were designed to amplify the sensor signals.

LabJack was chosen as a data acquisition device and LabVIEW software was used to acquire and analyse the signals transmitted by LabJack. Before analysing the signals, they were pre-processed for noise reduction and signal conditioning. The root mean square of the signal was found and used for analysis. The output voltages of the pressure sensor were converted to range of 4-20 as the pressure range of the PAP

machine is 4-20 cm of water. Also, as the amplitude of the audio sensor did not change much for leaks, the area under the curve was found and used for leak detection. The leaks are classified based on the location as Mask and Machine leaks.

The algorithm was only able to detect leaks of certain pressure ranges and it was not able to detect leaks of higher pressure range. Then the system was updated by replacing the sensors with an amplified and more sensitive one. The algorithm was also improvised by setting thresholds for each pressure range which increased the accuracy to 75%. The system was not still able to detect very fine leaks that occurs on the mask side.

Therefore, the system was again updated with new programmable Microcontroller called Arduino and a new programming software called MATLAB. The Arduino was programmed to transmit the signals using serial port communication. A Moving Average filter with window size 500 was designed to reduce the noises in the system and the filtered signal values were again filtered to obtain a distinct signal for each set pressure with no leaks and for all intensities of leaks. The algorithm, instead of using the pressure sensor values separately, used the difference between the machine side and mask side pressure sensor values for detecting the leaks. The audio sensor on the mask side was eliminated and the audio sensor that was previously installed inside the connector was moved to outside for better response. An accuracy of 100 % was achieved in Laboratory testing. A Graphical User Interface was designed to start and stop the software and choose the notification method. Two notification methods were designed namely text message and alarm. Based on the user's choice the corresponding notification method will be used to notify the user if the leak sustains longer than the period mentioned by the clinician.

Table of Contents

List of Illustration

List of Tables

1. Introduction

1.1. Sleep Apnea

Sleep depends on various factors like sex, environment, occupation, diet, age etc. The average sleep time for a normal human being is between 16 hours and 6 hours. The average sleep time for neonates is 16 hours a day which, decreases with increase in age and is 6 hours for extremely old age [2]. The living organisms follow a regular sleep pattern. The changes in this pattern is termed as sleep disorder. There are various reasons like excessive daytime sleep, irregular breathing etc that causes sleep disorder which may even be a threat for human life [2]. National Commission on Sleep Disorders Research that was established to study the sleep disorders and to develop a plan to use the national resources to deal with sleep disorders research reported that many Americans were disturbed by the sleep disorder symptoms [1].

Among various sleep disorders, the most common and serious one is Sleep Apnea. This disorder is characterized by periodic suspension of respiratory airflow during sleep. Sleep apnea is not a recent medical term. The symptoms of sleep apnea were observed 20 years ago, but unfortunately, having the key symptoms of this disorder as snoring, it was more considered as a humour than as a disease [1]. The symptoms of this disorder were mentioned by author Charles Dickens in one of his books named 'The Pickwick Papers' and thus the disorder was initially termed as Pickwickian Syndrome. Later, the first detailed polygraphic description of the manifestations of sleep apnea was provided by Gastaut in 1965 [5].

The various factors that contribute for sleep apnea are overweight, large neck size, small jaw etc. The risk is even increased by smoking and consumption of alcohol [3]. As previously mentioned, the sleep apnea is depicted as repeated cessation of airflow in the respiratory system during sleep [1]. This is caused by mechanical obstruction to the airway as well as by central lesions thus narrowing the pathway through which the air passes and not allowing the patient to get sufficient oxygen. Thus, as the patient tries to breathe through this airway, a loud snore is produced [2]. This disorder increases the carbon dioxide content and decreases the oxygen content in the human body [3].

Sleep apnea is classified based on the number of apneas per sleep time as, normal, which has less than 30 apneas and abnormal, which has more than 30 apneas per sleep time. Most patients have hundreds of apneas per night each lasting for 10 seconds. The number of apneas depends on sex, age and weight [2].

The sleep apnea is also classified based on the respiratory measurements as,

1. Central Apnea. No respiratory movements
2. Obstructive Apnea. The chest wall moves but there is no airflow at the nose or mouth.
3. Mixed Apnea. No Respiratory movements in the early episodes
4. Sub-obstructive Apnea. Reduced airflow with increased respiratory effort [2].

Among these, Obstructive Sleep Apnea (OSA) is the most common type of sleep apnea. The main consequence is excessive daytime sleepiness, snoring, sexual dysfunction and OSA is characterized by recurrent obstruction of the upper airway during sleep.

1.1.1. Obstructive Sleep Apnea

A sleep disorder caused by pharyngeal collapse during sleep resulting in complete or partial airway obstruction is termed as Obstructive Sleep Apnea. This type is classified as, hypopnea, in which the airflow is reduced and apnea, in which the airway is completely absent. The OSA usually has 15 or more episodes per hour of sleep [4]. The OSA decreases the oxygen saturation of the blood (Figure 1.1).

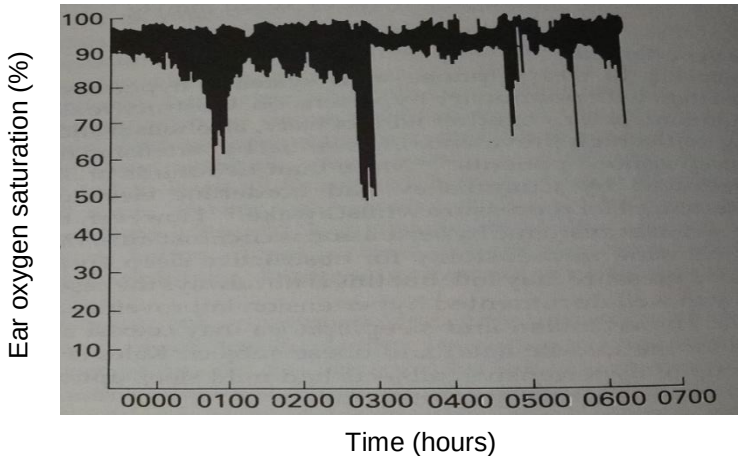


Figure 1.1 Record of oxygen saturation throughout the night in a patient with OSA [5]

The amount of fall of oxygen depends on the duration of apneas. Short apneas that last less than 20s can cause a fall from 95% to 80% while long apneas that last more than a minute can cause a fall of oxygen content up to 50%.

Studies have shown that OSA is more common in male than female. Also, the risk increases with age and it is more prevalent in older persons compared to middle aged individuals. Generally, patients with OSA are male, obese and are of age greater than 65 [4]. The prevalence of Sleep apnea is briefly explained in Table 1.1.

Table 1.1 Prevalence of OSA [6]

					Prevalence OSA		
			Number	Age (years)	AHI≥5	AHI≥10	AHI≥15	OSAS*
Young *et al.* 1993	USA	Men	352	30-60	24%	15%	9%	4%
		Women	250		9%	5%	4%	2%
Bixler *et al.* 2001	USA	Men	741	20-100			7%	4%
		Women	1000				2%	1%
Durán *et al.* 2001	Spain	Men	1050	30-70	26%	19%	14%	3,4%
		Women	1098		28%	15%	7%	3%
Young *et al.* 2002 (b)	USA	Men	2648	39-99	33%		25%	
		Women	2967		26%		11%	
Hrubos - Strøm *et al.* 2011	Norway	Men	284	30-65	21%		11%	
		Women	234		13%		6%	
Franklin *et al.* 2012	Sweden	Women	399	20-70	50%		20%	17%

Notes:
All studies used polysomnography to evaluate the AHI.

Definition of apneas:
Young *et al.* (1993), Bixler *et al.* (2001), Durán *et al.* (2001): complete cessation of airflow ≥ 10 s.
Young *et al.* (2002 b): 25% reduction in airflow for >10s accompanied by a 4% drop in SaO2
Hrubos-Strøm *et al.* (2011): 90% reduction in airflow >10 s.
Franklin *et al.* (2012): cessation of airflow for ≥10s.

Definition of hypopneas:
Young *et al.* (1993): reduction in airflow + decrease ≥ 4% in SaO2
Bixler *et al.* (2001): reduction in airflow ≈ 50% associated with a 4% reduction in SaO2
Dúran *et al.* (2001): 50 % reduction in airflow accompanied by a 4% reduction in SaO2 or EEG arousal
Young *et al.* (2002 b) and Hrubos- Strøm *et al. (2011)*: 30% reduction in airflow for > 10 s. with a ≥ 4% reduction in SaO2
Franklin *et al.* (2012): 50% reduction in airflow for ≥10s in combination with an arousal or 3% reduction in SaO2

*Definition of OSAS:
Young et al. (1993): OSAS = AHI ≥ 5 + daytime hypersomnolence =(≥ 2 days per week) of feeling excessively sleepy during the daytime, of waking up unrefreshed regardless of sleep length, and of uncontrollable daytime sleepiness interfering with daily activities.
Bixler *et al.* (2001): OSAS = AHI ≥ 10 + daytime sleepiness, hypertension or other cardiovascular complication. (a definition of daytime sleepiness was not described in the study)
Durán *et al.* (2001): *OSAS* = AHI≥10+ sleepiness ≥3 days/week during the past 3 months in one or more: after awakening, during free time, at work or driving, or during daytime in general.
Franklin *et al.* (2012): AHI≥5 + ESS score ≥10

OSA diagnosis is almost always preceded by progressive and excessive snoring. Snoring is a sound produced by the vibration of the soft parts of oropharyngeal walls, caused by transmission of part of the

energy of the inspiratory airflow and reflects in upper airway resistance [5]. In OSA, as the airway gets blocked by pharyngeal collapse, it is more likely that the patients with OSA tend to snore. Snoring is a common case affecting nearly 20% of the middle aged people and people with age over 60 tend to snore habitually. OSA patients are also unresponsive even to a painful stimulus [5]. As the patient doesn't get enough oxygen during sleep, they usually get morning headaches and abnormal movements during sleep. This disorder can even result in serious personal and professional problems [4].

In early stages, the patients wouldn't face much difficulties to sleep. Sedentary activities like watching television helps the patient to fall asleep. But later, the severity may increase due to increase in age, weight etc. and people find it difficult to sleep during night affecting their daily activities. The symptoms even vary with gender. Daytime sleepiness is more common in men whereas females experience symptoms like migraine headaches, irritable bowel movements etc. OSA not only affects the daily life, but it also tends to increase the risk of hypertension, diabetes, heart failure, coronary artery disease, stroke, arrhythmias, neurocognitive and mood disorders [4]. Also it is seen that sleep time of the patients with low numbers of apneas is not affected much whereas the patients with more number of apneas tend to sleep less [5].

Thus, Obstructive Sleep Apnea should be considered as a serious issue and should be diagnosed as soon as possible. There are many techniques developed to treat the OSA. Some of them are surgical treatment using radiofrequent energy, positive airway pressure therapy and Uvulopalatophalyngoplasty [5]. In the scope of this thesis, Positive Airway Pressure therapy will be discussed.

1.2. Positive Airway Pressure

The most common and effective method used for respiratory support is Positive Airway Pressure (PAP) therapy [1]. Positive airway pressure (PAP) uses a positive airway pressure ventilator. As the airway of the patient suffering from sleep apnea is blocked, the PAP machine can be used to keep the airway open by applying mild air pressure from the ventilator on a continuous basis.

The PAP machine consists of three parts namely the mask, a motor and a tube. The mask is tightly strapped over the nose or both mouth and nose of the patient. One end of the tube is connected to the motor and the other end is connected to the mask. The mask, tube and motor together forms a closed system [5]. On turning the machine on, the motor sends pressurized air of range 4-20 cm of water (0.0569-0.28 psi) to the patient, at high flow rates, through the tube. The machine is generally light and produces rhythmic noise. The setup of the PAP machine is shown in the below figure (Figure 1.2).

Figure 1.2 Basic PAP Setup [15]

In the above figure, the rectangular box is the machine that has motor to send pressurized air that goes through the tube connected to the mask and reaches the nasal cavity through the mask strapped around the patients' face. The pressure for the patient is determined by the clinician based on various factors like the apnea- hypopnea index, body mass index, etc. of the patient. The patient inhales the pressurized air through the mask and the pressure helps the airway to remain open and helps the patient to breathe without difficulties. The mask also has small holes to provide room for exhalation [2].

The goal of the machine is said to have achieved if it removes the airway occlusion and also, if it improves the sleep level of the patients and reduces effects like daytime sleepiness. thereby improving the quality of life of the patients. The achievement of goal is measured by the improved quality of life. In order to determine this, follow up experiments were conducted by researchers [10].

Many scientists and researchers have done experiments with the PAP machine and also have tested its working against different patients of different apnea index. A researcher conducted an 8-week PAP therapy

session with 6 sessions per week with variable duration and pressure for each session. It is proven that the increased air pressure delivered to the nasal cavities provide the weight against which the muscles of pharyngeal closure must work [6]. It was also found to have stabilized the blood pressure levels and have reduced the cardiovascular mortality [7].

Although it is an effective treatment for sleep apnea, common side effects and complaints of the patients become a challenge for its efficacy. The PAP machine has removed the physiological problem like apneas during sleep as well as has reduced the associated symptoms. But the patients seemed to experience some common side effects [8]. The intolerance and compliances are usually seen when the patients use the machine for a long duration [9].

The patients generally complain that it has physical side effects like causing skin allergy, nose congestion, sinusitis etc. They even seem to cause stomach bloating, discomfort during inhalation, nausea, vomiting and skin breakdown. Survey was taken on patients' experience with the PAP machine and it has also been observed that some people even experienced runny nose and claustrophobia [11].

The most common and important reason for its non-compliance is air leakage from the mask. There might be various reasons behind the air leaks. It might have been caused because of the material used in manufacturing the mask. Also, as the machine is mostly used during sleep, it is not possible for the patient to know when the mask moves from its position or if the tube or strap comes off. [10]. Thus, air leakage occurs and it remains active until the patient wakes from the sleep and readjust it.

It is also seen from the patients' feedback that not only the mask movement causes the leaks but also the PAP machine itself causes it. The most commonly reported problem with the equipment is that the straps attached to the mask is not of the right size for it and stands as one of the reason behind the mask leak. The leaking mask will also result in causing the previously mentioned side effects such as nose congestion, sinusitis and nosebleeds [10].

1.2.1. Resmed

The PAP device used for this project is Resmed AirSense 10. This is an intelligent device that follows the common PAP setup with a device that has motor to send pressurized air, a tube through which the air

passes and finally reaches the mask that is tightly strapped around the patients' nose. The image of the Resmed AirSense 10 device is given below in Figure 1.3.

Figure 1.3 Resmed AirSense 10 PAP Machine
source: Resmed AirSense 10 Datasheet

The above figure also indicates the purpose of each part of the device. The device comes with a power connector to power up the device. The rear of the device shown in the below figure (Figure 1.4) has a power jack to which one end of the power connector would be connected and the other end goes to the power outlet. The device turns on only after it receives power from the outlet. The rear of the device also has an air outlet to which the tube would be connected and the air from the device passes through the tube.

Figure 1.4 Rear view of the Resmed PAP machine
source: Resmed AirSense 10 Datasheet

The device has a display in the front. On turning on the device, it displays the home screen with two main options namely My Options and Sleep report. The dial next to the display can be used to navigate between the options. While navigating, the options get highlighted and on pressing the dial selects the corresponding option. The 'My options tab' is shown in the Figure 1.5 given below

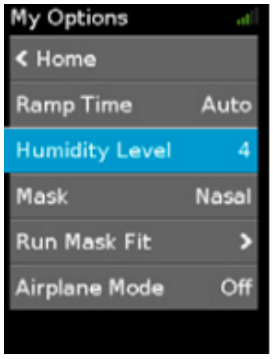


Figure 1.5 My options' menu of Resmed PAP machine
source: Resmed AirSense 10 Datasheet

Usually a constant pressure that is required to open the patient's airway is set by the clinician, but if the patient finds it difficult to handle the same pressure continuously or if they experience a bloated feeling, the ramp time can be set. If the ramp is off, on starting the therapy, the pressure rises within few seconds and reaches the set pressure. On setting the ramp time to a particular time, say m minutes, the pressure increases from the start pressure proportionally based on the time that is being set and reaches the set pressure at the m^{th} minute. If the patient feels that they aren't receiving enough air, the ramp time can be turned off.

The humidity can also be set based on the comfort of the patient. If the patient experiences a dry or runny nose, the humidity level can be increased and on contrast, if they experience water droplets in their nose, mask or air tubing, the humidity level can be decreased. The Run mask fit test can be performed to ensure the mask properly fits and the system has no leaks. On selecting the 'Run Mask Fit', runs a trial of the therapy and checks for the mask fit and indicates it to the user. The device has an option to connect to the WiFi network. It is used to upload the summary of each therapy to the internet, so as to allow the clinician to view the details about the therapy underwent by the patient. To disable this facility, the 'Airplane Mode' option is selected. Again to choose each option in the My Options tab, the dial is used. The device also has a slot to insert an SD card, so that the raw data after each therapy session would be uploaded to the card

in the '.edf' format. The data gets arranged in a folder based on the date and also, the filename has the time of the session included in it. To open the '.edf' file, a software called EDF browser is used. This software has an option to convert the edf format file to ASCII file, which can then be opened using an excel spreadsheet and the raw data is seen. The signals acquired from the device is shown in the figure given below (Figure 1.6).

Time	MaskPress.2s	Press.2s	EprPress.2s	Leak.2s	RespRate.2s	TidVol.2s	MinVent.2s	Snore.2s	FlowLim.2s	Crc16

Figure 1.6 Signals obtained from Resmed PAP machine

From the data, for the scope of the project, only the time, MaskPress.2s, Press.2s and Leak.2s columns are used. The time column represents the differential time from the start till the end of the therapy in steps of 2 seconds. All the signals are recorded every two seconds. The 'MaskPress.2s' represents the actual pressure that is observed in the system and the 'Press.2s' represents the actual pressure that is being set. The unit of pressure is centimetre of water. The 'Leak.2s' represents the amount of leak in units of Litre per second.

1.3. Leak Detector System

Due to the occurrence of leaks, the patient does not receive sufficient pressure to open their airway or even sufficient air at all, thus making the system and treatment ineffective. To overcome this issue, the user should be alerted or notified of the occurrence of the leaks so that they can readjust the mask or hose to be completely benefited by the therapy. The PAP machine provides information about the pressure and leaks but only after the machine is stopped and does not alert the user during the run. Thus this project focusses on developing a leak detector system for the PAP machine and indicating the time and location of the leaks to the user as it occurs.

1.3.1. Functionality of the System

There is a possibility for the leaks to occur either at the location at which the hose is connected to machine or to the mask and also near the mask seals due to the movement of the mask away from the nose.

The main aim of the project is to develop a system that senses whenever a leak occurs while using the PAP machine and indicate the time of occurrence of leak and also indicate whether the leak is at the mask

side or at the machine side. The additional feature of the system is to allow the user to control the method of notification of leaks. The leaks can be notified either by turning on an alarm or by sending a text message or e-mail to alert the user.

The scope of this thesis is sub divided into following tasks.

1. Identify appropriate sensors for detecting leak that meet the requirements of the PAP machine operation
2. Design electronic circuit for interfacing between the sensors and the computer software program
3. Develop the software algorithm for determining when and where leak has occurred based on the sensor input
4. Develop the user interface for operating the leak detector system and viewing the data
5. Develop multiple methods for notifying the user of when and where the leaks are occurring

2. Leak detection Approach – Version 1.0

2.1. Sensor Selection

Many scientists and researchers have already done research on PAP machines and their associated leaks [13]. Various approaches are already in practice to detect air, water and other gaseous leaks while flowing through a pipeline [14]. General approach is using sensors like pressure sensor, flow sensor, audio sensor etc. This project also follows the general approach of using the sensors. Since the system deals with pressures and as the pressure changes based on the leaks, pressure sensor was considered as a prominent sensor. The pressure sensor was sufficient to detect if there is a leak but to determine if the leak occurs at the machine or mask side, another sensor was required. Audio sensor was chosen to be another ideal sensor as it reacts to the sound caused by leaks. Based on the response of the sensors the existence and intensity of the leaks are determined.

2.1.1. Pressure Sensor (PS)

As previously mentioned, pressure plays an important role in the device, choosing the appropriate pressure sensor is the primary task. So, research was done on the various types of pressure sensors, of which some

are listed in Table 2.1 given below. It was found that an appropriate sensor would be Honeywell TBPDANS001PGUCV, which is an unamplified, compensated sensor, having a single axial barbed port and measures gauge pressure. This sensor was chosen because it was designed for air leak detection and has an accuracy of (+ or -) 0.20 %. The sensor is also capable of handling 0 to 95% humidity.

Table 2.1 Other Pressure Sensors evaluated

S.No	Model	Company	Accuracy % (+ or -)	Comments
1	SSC Series	Honeywell	2	Used in respiratory environment
2	HSC Series	Honeywell	1	Used in respiratory environment
3	CXLDP, DXLDP, IXLDP	Ashcroft	0.4 – 0.8	Did not have ports to fix it to the connectors
4	TBPDANS001PGUCV	Honeywell	0.2	Used commonly for air leak detection

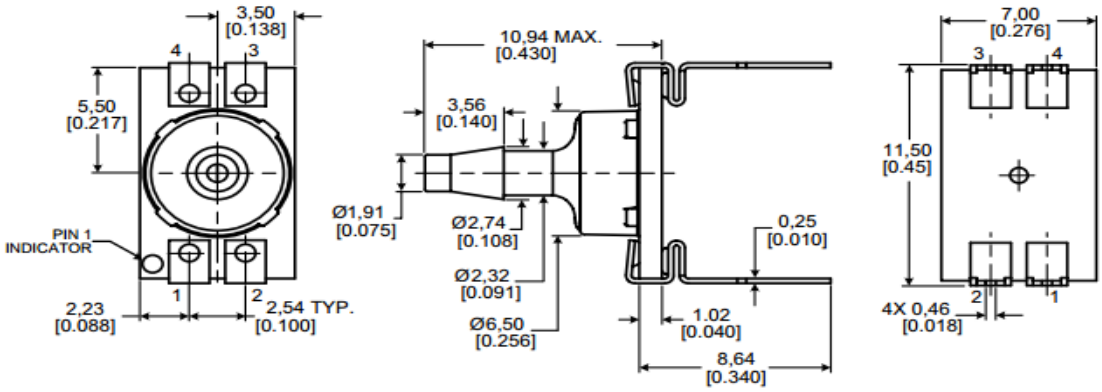
The side view of the sensor is given in the figure below (Figure 2.1).

Figure 2.1 TBPDANS001PGUCV Pressure Sensor Side View
Source: TBPDANS001PGUCV datasheet

The pressure sensor has four terminals (1-4). Terminal 1 and 3 are Supply and Ground Connections respectively. The typical voltage required is 5V with respect to Ground. The Terminal 2 is negative output voltage ($Vout^-$) and the Terminal 4 is positive output voltage ($Vout^+$). A hole is provided at one end of the sensor for identification of pins. The pin close to the hole is Pin 1 and increases in the anti- clockwise direction. The dimensional drawing and pin out is given in the below figure (Figure 2.2).

The Pressure sensor is capable of measuring pressures of range from 0 to 1 psi (0 – 70 cm of water). The Pressure range specifications of the pressure sensor is given in the table (Table 2.2) below.

The sensor also reacts to the changes in temperatures. The thermal effect on offset is at the maximum of ± 2.05 % of the FSS and the thermal effect on span is at the maximum of ± 2% of the FSS. This is a high quality pressure sensor and also cost effective, flexible, durable and reliable.



Pin 4	Pin 3
Vout+	GND
Pin 1	**Pin 2**
Vsupply	Vout-

Figure 2.2 Dimensional drawings and pinout of TBPDANS001PGUCV Pressure Sensor
Source: TBPDANS001PGUCV datasheet

Table 2.2 Pressure Sensor TBPDANS001PGUCV Specifications

Pressure Range		Unit	Full Scale Span (FSS) (units: mV/V)			Offset	Over Pressure	Burst Pressure	Accuracy
min	max		Min	Norm	Max				
0	1	psi	1.42	1.50	1.61	± 0.075	12.7	20	± 0.20

2.1.2. Audio sensor (AS)

The pressure sensor alone was not sufficient in detecting the leaks. So, in order to assist the pressure sensors, it was planned to use one more sensor along with them. Audio sensor was chosen for this purpose as the leaks produce hissing sounds when they occur, based on which, the audio sensor reacts. Thus various audio sensors listed in Table 2.3 were analysed and finally, POW-1644L-LWC50-B-R audio sensor manufactured by PUI Audio was chosen as it was sensitive enough to detect changes in sound associated with leak and also because it is not affected by the human voice or music. Another reason for choosing these sensors is that they can resist the humidity caused while the patient exhale and the humidity caused by the humidifier in the machine.

Table 2.3 Other Evaluated Audio sensors

S.No	Model	Company	Comments
1	AKU 442	Akustica	Not suited for mounting in the hose
2	MP33AB01H	ST	Accepts max input voltage of 3.6V. Not compatible with LabJack voltage
3	POM-3535L-3-R	PUI Audio	Not suited for high humidity environment
4	ADMP441	Analog Devices	Not suited for mounting in the hose
5	POW-1644L-LWC50-B-R	PUI Audio	Suited for humid environment

The side view of the audio sensor is given below in Figure 2.3.

Figure 2.3 POW-1644L-LWC50-B-R Audio Sensor Side View
Source: POW-1644L-LWC50-B-R datasheet

The red wire in the above figure (Figure 2.3) is the positive terminal and the black wire is the negative terminal. The operating voltage range of the audio sensor is 2 – 10 V. The operating frequency range is 50 to 16000 HZ and has a sensitivity of -44dB. The dimensional drawing of the audio sensor is given below in Figure 2.4.

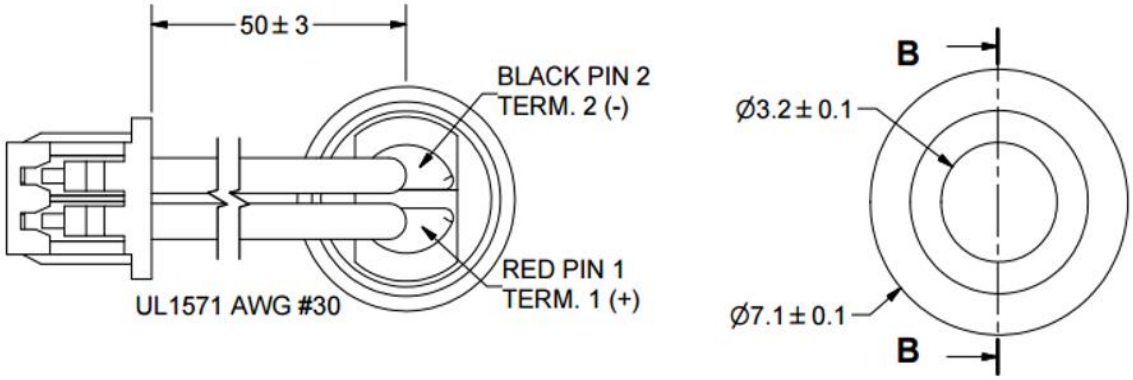


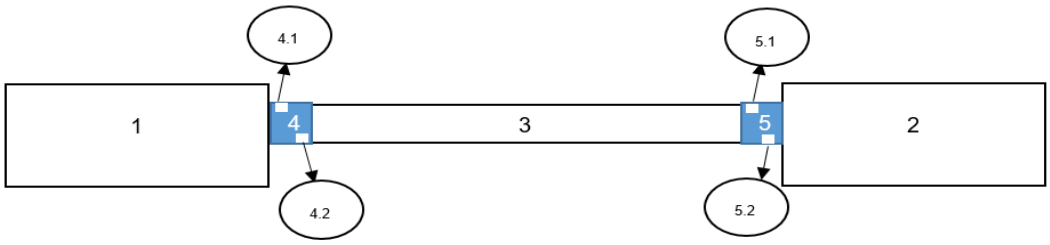
Figure 2.4 Dimensional drawings of Audio Sensor POW-1644L-LWC50-B-R
Source: POW-1644L-LWC50-B-R datasheet

The output voltage is in order of few millivolts. The connections for the audio sensor will be explained in Section 2.4.1.

2.2. Placement of Sensors

One of the key factors after determining the type of sensors is to properly place them to detect the leaks accurately. Two pressure sensors and two audio sensors were chosen for use. The mask side sensors are numbered as 1 and the machine side sensors are numbered as 2. A combination of a pressure sensor and an audio sensor (PS1 and AS1) was installed near to the mask in between the mask and the hose and another combination of pressure sensor and audio sensor (PS2 and AS2) was installed near the machine

between the hose and the machine. Two connectors were designed to hold the two pairs of sensors. The connectors are designed in such a way that the sensors are placed inside the connectors in the path where air flows. Initially, a temporary connector was designed using a rod and a plastic tube. The tube and the rod were selected based on the diameter of the end of the hose. Holes were drilled on the connectors based on the diameter of the sensors given in the previous sections (2.1.1 and 2.1.2) and the diaphragm of the sensors are inserted into the connector. The sensors and connectors setup is given below in Figure 2.5. The connectors were then updated to a single piece using Solid works software and holes were drilled for the sensors. One end of the connector was designed based on the diameter of hose and other end of the connector was designed based on the diameter of mask and machine end connectors. This was done by one of my team mate Castro Jesus De Jose. He was pursuing his Bachelors in Biomedical Engineering and is working as an Intern at UTARI. The details about the connector designs are explained in the Appendix B.



1 – CPAP Machine
2 – Mask
3 – Tube
4 – Connector for Mask side
1.1 – Pressure Sensor (PS1)
1.2 – Audio Sensor (AS1)
5 – Connector for Instrument Side
5.1 – Pressure Sensor (PS2)
5.2 – Audio Sensor (AS2)

Figure 2.5 Connectors and Sensors setup

2.3. Interface

After mounting the sensors, they need to be powered up for operation and their output voltage which is analog in nature, should be measured in order to design an algorithm for detecting leaks. Algorithm can be

designed using a programming software in a computer. But, unfortunately, it is not possible for the computer to directly read an analog signal. An interface is required to send the sensor signal to the programming software. LabJack is chosen to serve this purpose. A LabJack is a data acquisition device with analog and digital input and output ports. There are many types of LabJack available out of which LabJack U3 – HV is chosen (shown in Figure 2.6) for the project as it is the simplest and well suited for analog read operations.

The AIN0 – AIN3 in Figure 2.6 are the analog inputs. The VS port gives an output voltage of 5V with reference to GND port. The VS and GND can be used to power up the sensors. The sensor outputs can be connected to the analog ports. The LabJack has an analog to digital converter (ADC) to read the analog ports and convert them to digital values to allow the computer to read it. The LabJack has a 12-bit ADC. The detailed description of the ADC is given in the following subsection (2.3.1). The LabJack communicates the sensor voltages with the computer via a Universal Serial Bus (USB). The USB cable from the LabJack is connected to the port where a flash drive will be connected in the computer.

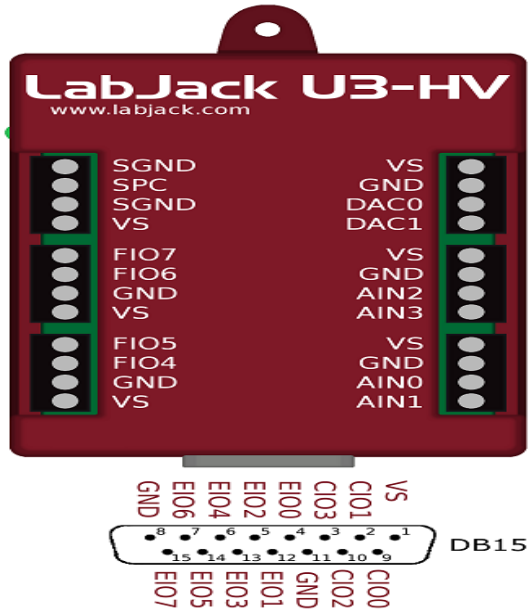
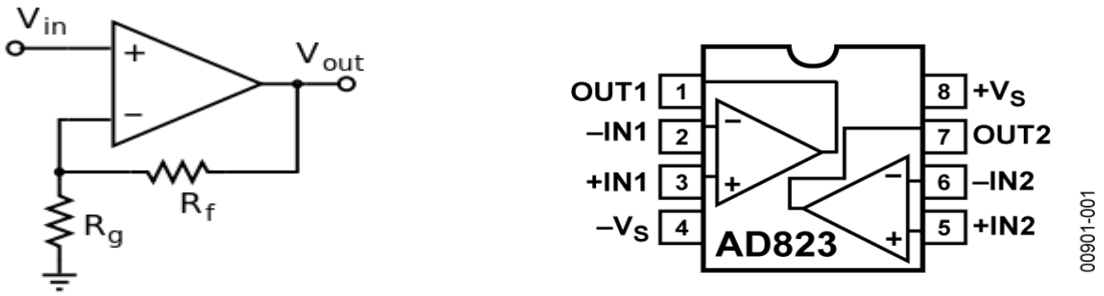


Figure 2.6 LabJack U3 – HV
Source: LabJack datasheet

2.4. Circuit Design for Sensors

The LabJack provides an easy and user friendly interface. The bottleneck here is that the LabJack can accept voltage only in the order of volts (0-5V) and the output of the selected sensors are in order of few

millivolts. It wouldn't be possible for the LabJack to differentiate any amplitude changes for those very low voltages. So, an additional circuit is built to amplify the sensor output so that it can be read by the LabJack. An operational amplifier (OP-Amp) Integrated Circuit (IC) is used for amplification. The Op-Amp chosen for the project is AD-823 IC. The basic schematic of the Op-Amp and the pin layout of AD823 is given in below figure (Figure 2.7)



(a) (b)

Figure 2.7 (a) Schematic of basic Op-amp (b) Pin layout of AD823 Op-Amp IC
Source: AD823 datasheet

Given above is a non-inverting amplifier where R_f is the feedback resistor and R_g is the gain resistor. The Op-amp IC in Figure 2.7(b) has 2 Op-amps in it. The -IN is the inverting terminal and +IN is the noninverting terminal. The Op-Amp also requires +5V to be connected to V_S and -5V to be connected to the $-V_S$. The LabJack is capable of providing positive voltages only. Additional components will be required for providing -5V for the Op-amp, the details of which is discussed in the following subsection (2.4.1). The gain of the amplifier depends on the resistors used. The gain calculated using the formula is given below.

$$Gain = \left(1 + \frac{R_f}{R_{in}}\right)$$

And the output voltage is

$$V_{out} = Gain * V_{in}$$

The circuit design is different for each sensor. The description of the circuit design for each sensor is explained in the following subsections.

2.4.1. Charge Pump

As mentioned earlier, the Op-amp requires -5V for power up and as LabJack provides only +5V, a charge pump circuit is required to provide the negative voltage. The charge pump IC ICL7660S is chosen for the project. It acts as a voltage converter. It requires an input voltage of 5V and the circuit converts and gives an output of -5V. The schematic of the charge pump circuit is given below in Figure 2.8. The supply current associated with the supply voltage is taken and divided into half in which one half goes directly to the positive side of the load and the other half goes to the negative side of the load. Therefore,

$$V_{out} \cong 2V_{in}$$
$$I_s = 2I_L$$
$$V_{in}I_s = V_{out}I_L$$

Where,

V_{out} is the output voltage,

V_{in} is the input voltage,

I_s is the supply voltage and

I_L is the load current.

The ICL7660 IC along with two external capacitors act as a negative voltage converter circuit. The mode of operation is shown in the figure (Figure 2.9) given below. The time cycle is divided into two. The circuit uses 4 MOS power switches in which S1 is a P channel switches and S2, S3 and S4 are N channel switches. For the first half of the time cycle, the Capacitor C1 is charged to the supply voltage. At this time, the switch S1 and S3 are closed and S2 and S4 are kept open, thus providing an output voltage of +V volts at pin 8. During the second half of the cycle, the process is reversed, the switches S1 and S3 are kept open and switches S2 and S4 are closed. Therefore, the capacitor is shifted negatively by V+ volts. Charge is then transferred and maintained by capacitor thus providing an output of -V volts at pin 5.

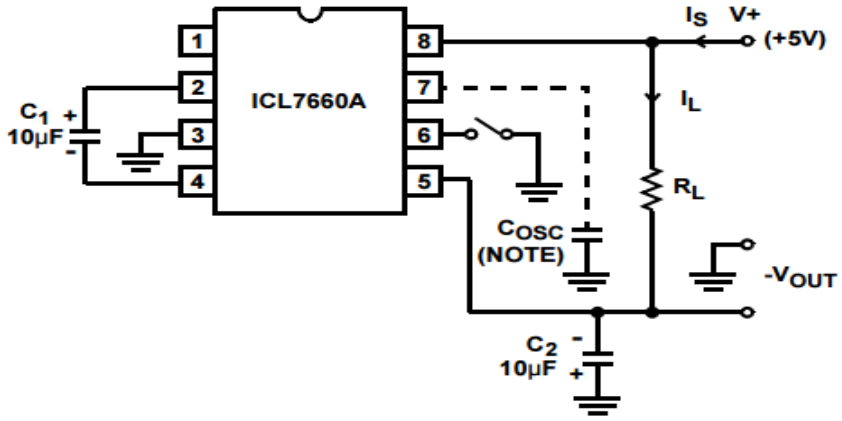


Figure 2.8 Schematic of charge pump circuit
Source: ICL7660A datasheet

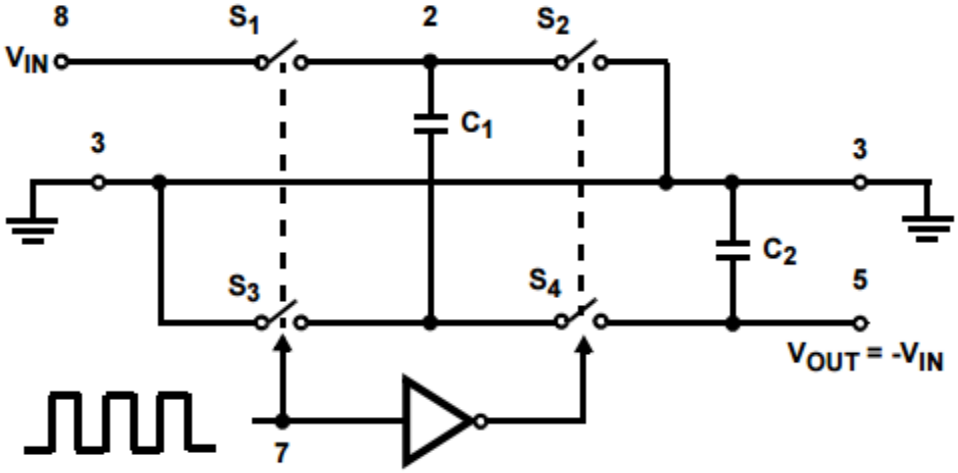


Figure 2.9 Ideal Negative Voltage Converter
Source: ICL7660A datasheet

2.4.2. Pressure Sensor

It is found that there is an offset between terminal 4 and terminal 2 of the pressure sensor. This offset is known as null offset. If the null offset is not eliminated, a negative null offset could maintain a zero voltage for very low pressure ranges. To eliminate this, a circuit is designed which inserts a resistor between the Terminal 1 and Terminal 4 of the sensor and maintains the offset to a positive value. The null offset circuit is given below in Figure 2.10. The Various amplifier circuits were designed by Honeywell, the manufacturer

of PS, for their pressure sensors. The circuits were carefully studied and an optimal one is chosen. As seen in section 2.1.1, the pressure sensor gives both positive and negative voltage.

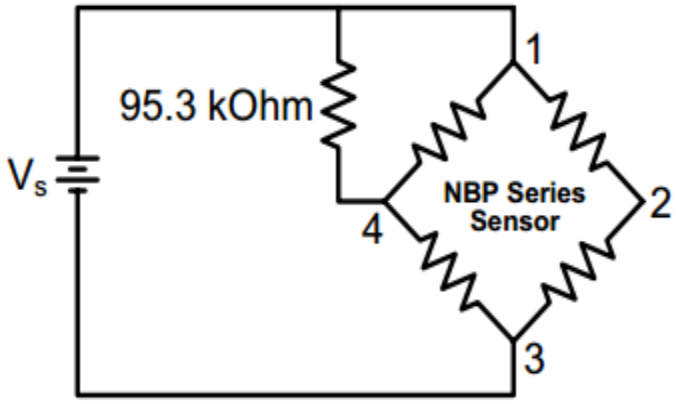


Figure 2.10 Null offset Circuit
Source: Honeywell output signal adjustment technical note

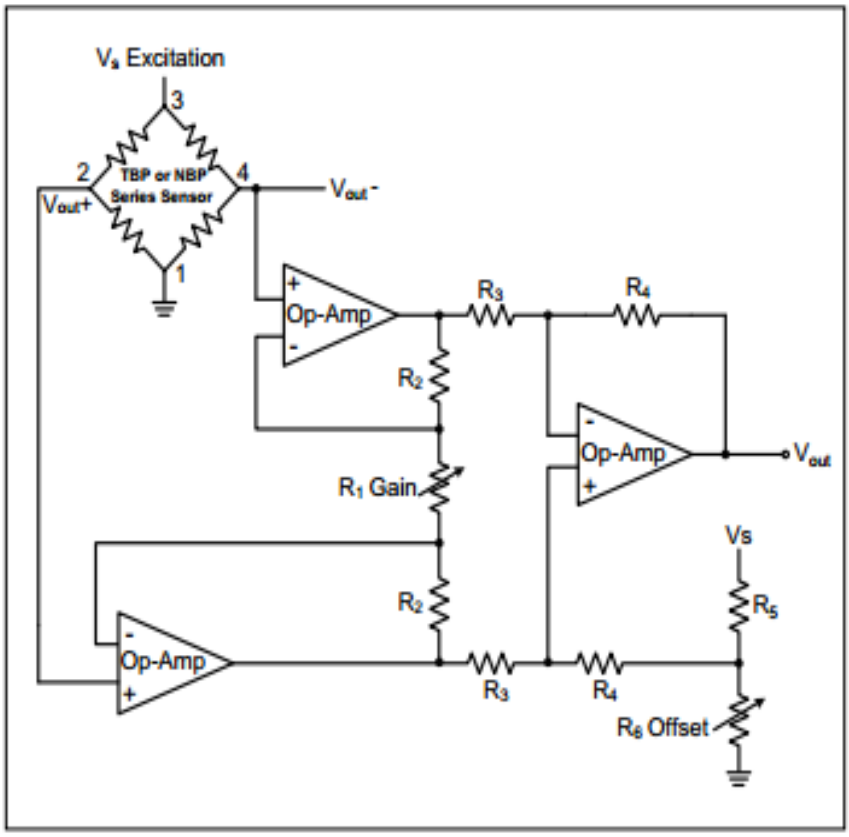
Therefore, an instrumentational amplifier circuit is found to be the optimal choice. An instrumentational amplifier is a combination of three operational amplifier circuits. The $Vout^+$ of the pressure sensor (Terminal 4) is given as input to one Op-Amp circuit and $Vout^-$ of the pressure sensor (Terminal 2) is given as input for the other Op-Amp Circuit. The second input of the two Op-Amps are connected through a resistor R_1. The gain of the two Op-Amps are decided by this resistor. R_2 acts as a feedback resistor for the two Op-Amps. The amplified output of these two Op-Amps are connected to the two inputs of the third Op-Amp Circuit through resistor R_3. R_4 acts as the feedback resistor of the third Op-Amp. The schematic of the Pressure Sensor amplifier circuit is given below in Figure 2.11.

The equation of the final output voltage is

$$V_{out} = \left(\left(2\frac{R_2}{R_1}\right) + 1\right)\left(\frac{R_4}{R_3}\right) + V_{offset}$$

Where,

$$V_{offset} = V_s * \left(\frac{R_6}{R_5} + R_6\right)$$



R1 = 20 kOhm potentiometer	
R2 = 100 kOhm	
R3 = 100 kOhm	R4 = 200 kOhm
R5 = 100 kOhm	
R6 = 50 kOhm potentiometer	

Figure 2.11 Schematic of Pressure Sensor Amplifier Circuit

2.4.3. Audio sensor

As the output voltage of the audio sensor is very low (typically in the range of few millivolts), the audio sensor output should be amplified before connecting to the LabJack. Additional components are required for the audio sensor amplification circuit. The table (Table 2.4) given below contains the components required for the audio sensor amplifier circuit.

Table 2.4 Audio sensor amplifier circuit

Components	Vendor Part #	Quantity
10K ohm resistor	71-RN55D-F-10K	1
56.2K ohm resistor	71-RN55D-F-56.2K	2
38K ohm resistor	71-RN55E3802B	1
47-ohm resistor	71-RN55D47R0F	1
10 µF capacitor	667-ECA-1HHG100	3

Given below in Figure 2.12 is the amplifier circuit for the Audio sensor. As mentioned in Section 2.1.2, the audio sensor has a positive as well as a negative terminal. Before connecting the positive terminal, also referred as Audio Sensor Output, to the circuit, it is connected directly to 5V supply. The negative terminal is grounded. To prevent the audio sensor from interfering with the DC components, the positive terminal is connected to a 10µF capacitor. Also to bias the audio sensor, the output of the sensor is connected to a 10K ohm resistor. This is then connected to the non- inverting terminal of the op-amp. The inverting terminal decides the gain of the circuit. The resistors chosen were 38K (R_f) and 15K (R_1), so as to obtain an output voltage that is compatible with the LabJack.

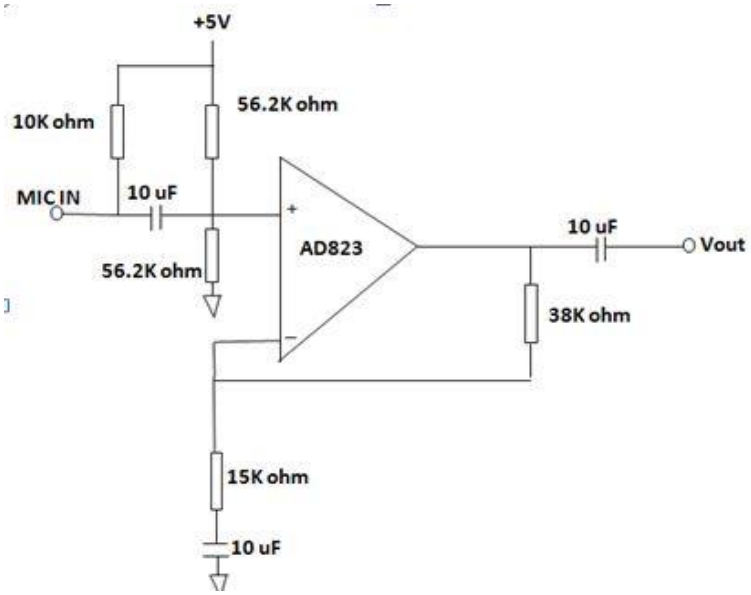


Figure 2.12 Audio Sensor Amplifier Circuit

The circuits were soldered on to a prototyping board and based on the dimensions of the circuit board and LabJack, a box was chosen to encase both. The image of the box is given below in **Figure 2.13**

Figure 2.13 Circuit Box

The box is designed to provide space for inserting the USB cable to the LabJack and plug in the sensor wires to the circuit. The image of the Leak detector system is given below in **Figure 2.14**

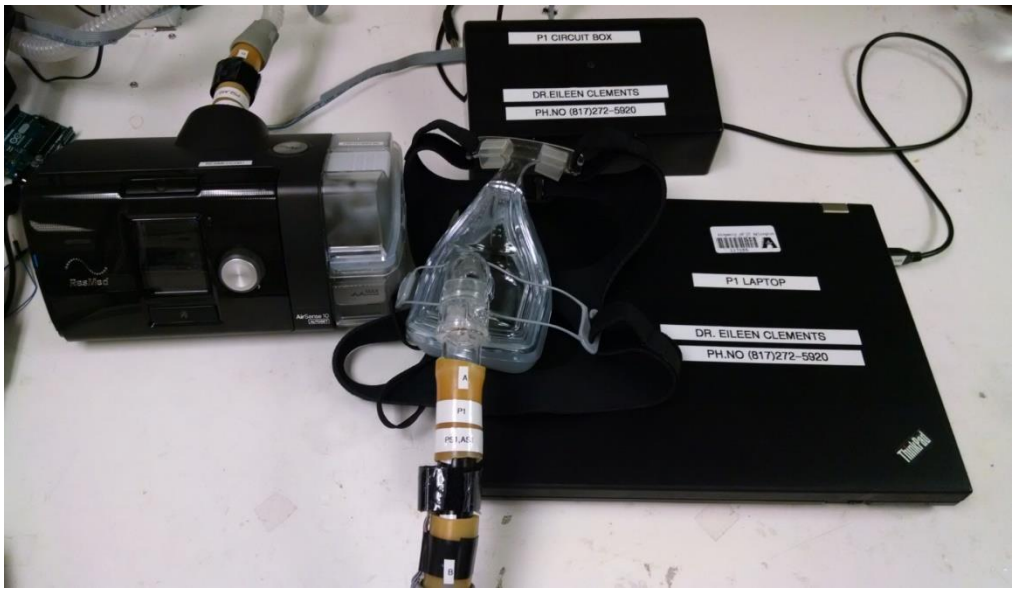


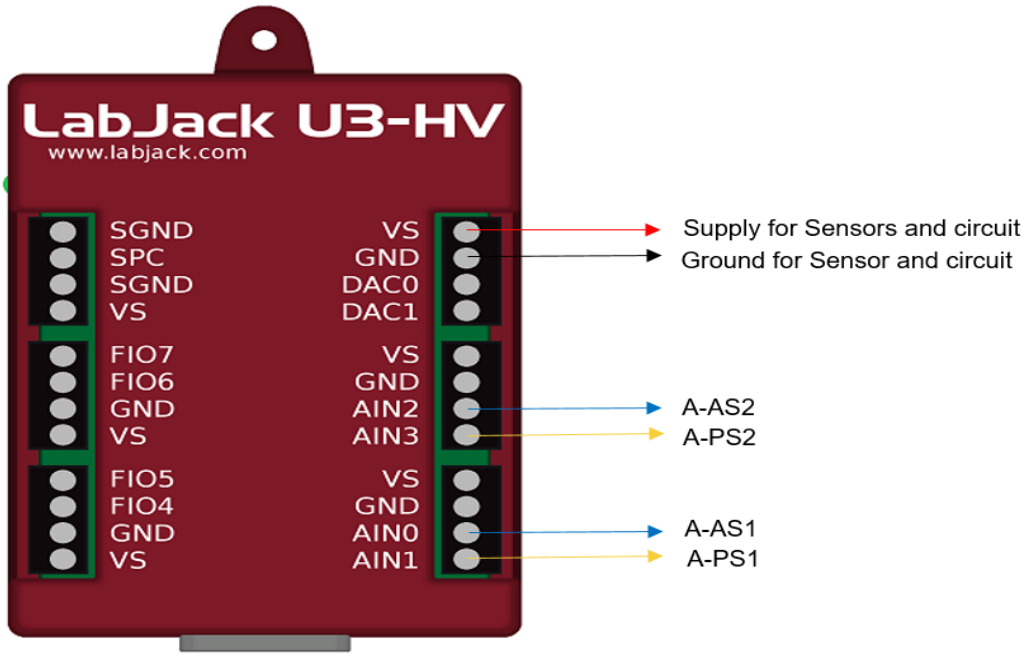
Figure 2.14 Leak Detector System Setup

3. Software

3.1. Data Acquisition

The sensors' outputs are amplified and these amplified outputs are connected to the analog input ports of LabJack. The connections to the LabJack are given below in Figure 3.1. The LabJack then has to be connected to the computer and a programming software has to be chosen to work with the data. LabVIEW is chosen to start with since it is easy to work with and has graphical icons and blocks that can be wired to perform certain operations. It has 2 windows, the front panel and the block diagram. The LabVIEW has a start button and a stop button to start and stop the program. The front panel consists of icons for inputs and outputs and the block diagram consists of icons for performing the functions with the inputs to obtain the necessary outputs. The main task of the algorithm are as follows,

1. Acquire the data from LabJack
2. Analyse the pressure and audio sensor changes for leak and no leak conditions
3. Differentiate between the two conditions
4. If the leaks sustain for a period of time (which will be given by the clinician), the software has to alert the user about the leaks either by text message or by ringing an alarm



A-AS1 – Amplified AS1 output

A-PS1 – Amplified PS1 output

A-AS2 – Amplified AS2 output

A-PS2 – Amplified PS2 output

Figure 3.1 Connections to LabJack

The LabJack generally operates in 2 modes: command response mode and Stream mode. Based on the application for which the LabJack is used and the time interval required between 2 samples, the corresponding modes are chosen. Command response mode is simpler and allows maximum scan rate of 1000 Hz. Stream mode streams the data from all the specified ports at a specific rate and places the samples in a buffer until next set of sample occurs. Thus the amplified sensor values are streamed at a scan rate of 10 KHz. As the program is executed, the software goes to wait mode and waits for the samples

specified in the scan rate to arrive. Once the samples arrive, the sensor value read by each channel is stored in its corresponding array. These values are then used for further operation.

3.2. Signal Processing

To continuously acquire the signal, a while loop is used. The loop executes the tasks inside the loop repeatedly until the condition given for stopping the loop is met. To stop the loop, a stop button is connected as the condition and the loop runs until this button is pressed. An interval can be specified for the loop to wait before repeating the tasks. Thus, the signal acquisition, processing and analysis are given inside the loop. The signals obtained from the sensors has noise in it. To eliminate the noise, the signal needs to be manipulated to obtain meaningful information about the change in signals for pressure changes and leaks. Also there wasn't much deviation in the pressure sensor output for changes in pressure. The pressure sensor output voltage corresponding 4 to 20 cm of water is 1.858 to 1.88V. With noises, it was difficult to differentiate the pressure changes and also for the audio sensor, there wasn't much change in amplitude when a leak occurs. To overcome these, the root mean square of the signal is found. For the pressure sensor, the mean of the array is calculated each time the array is formed from the samples, and for the audio sensor, the values are integrated over time and the area under the curve is calculated. The area under the curve for mask side audio sensor is named as AS1 and for machine side audio sensor as AS2. Similarly, the mean value for mask side pressure sensor is named as PS1 and for machine side as PS2. Figure 3.2 below shows the change of pressure sensor output voltage for a change of pressure from 4 to 20 cm of water.

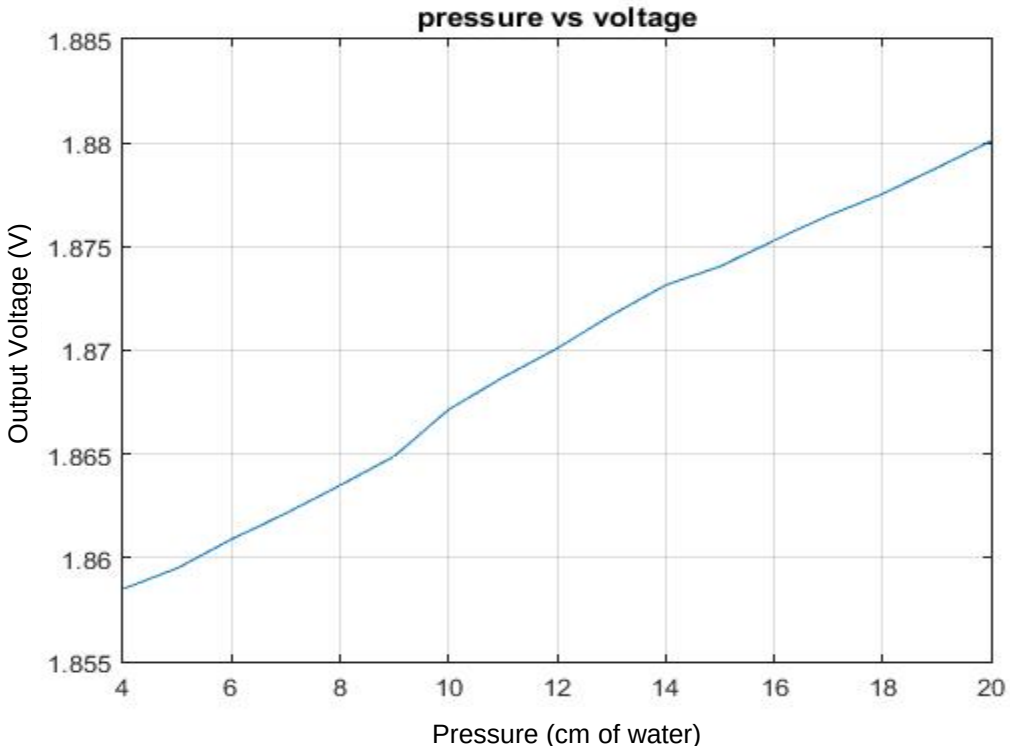


Figure 3.2 Output voltage (V) vs Set pressure (cm of water)

The output voltage of the pressure sensor is found to have a linear relationship with the change in input pressure. Therefore, the output voltage ranging from 1.858 to 1.88 is converted to 4 – 20 cm of water using simple math conversion with 1.858 representing the value 4 and 1.88 representing the value 20. The conversion is done using the formula

$$\frac{x - x_1}{x_2 - x_1} = \frac{y - y_1}{y_2 - y_1}$$

Where, x is the pressure sensor output voltage and y is the corresponding value in 4 – 20 range. Before this conversion, the averaging of the pressure sensor voltage is done and the average is then converted to 4 – 20 range.

The pressure was manually increased in steps from 4 to 11 cm of water and the responses before and after averaging are plotted. The difference between the response of the pressure sensor before averaging and after averaging is clearly seen in the below figures (Figure 3.3 and Figure 3.4). It is seen that the noises have reduced after averaging.

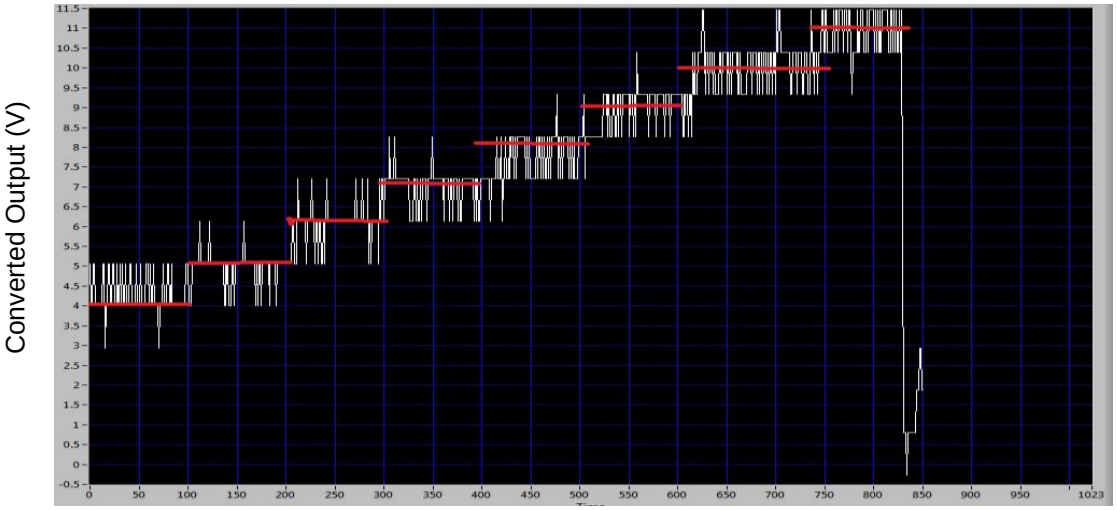


Figure 3.3 Output graph for variations in pressure before averaging

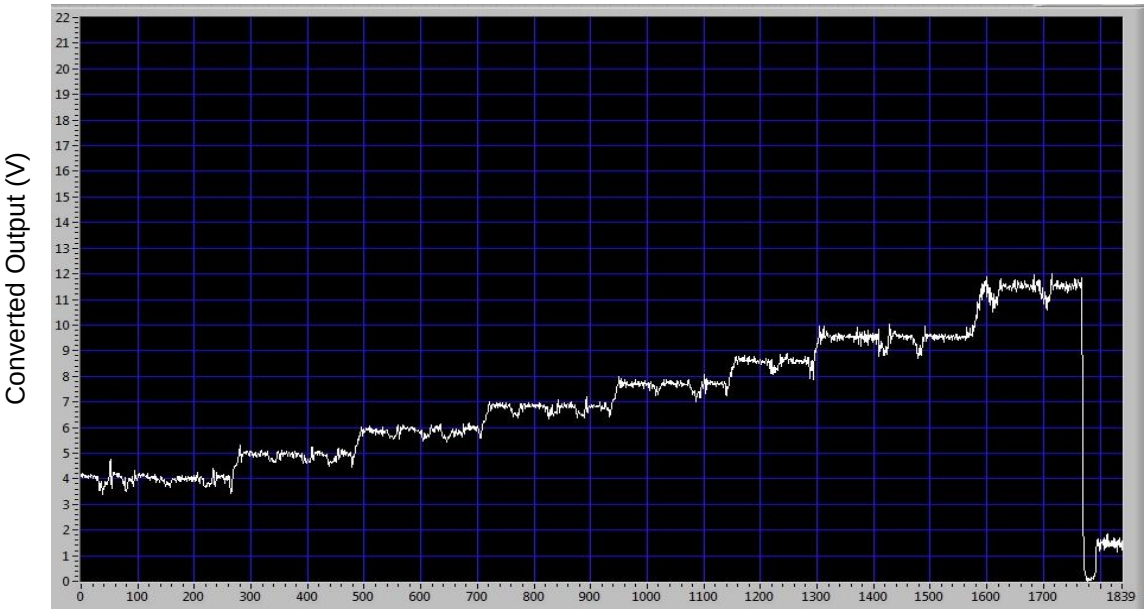


Figure 3.4 Output graph for variations in pressure after Averaging

3.3. Data Analysis

For analysing the data, the system is first programmed to export the AS1, AS2, PS1 and PS2 values to MS excel every 2 seconds after the start of the software until it is stopped. The 2 second interval is chosen because the PAP machine records the data every 2 seconds. The machine and the software is run and various tests are performed for leaks. The data collected from the software is compared against the leak

column of the PAP machine data (shown in Fig 1.6). The leak column of the machine data varies from 0 to 2 with 0 representing no leaks and 2 representing 100% leak. The columns are matched based on their corresponding time. This comparison is done to see the sensors' variation when the leak column indicates value greater than zero. Based on the inferences, the algorithm is designed. The machine was run for a certain period of time and leaks were purposefully created by moving the mask away from the face. This is the mask side leak. The data collected from the software during the test run is given in the Table 3.1 below. Also the data from the machine is collected. The leak column of the machine data is added to the table. The first four columns are the data exported from the LabVIEW programming software representing the sensor values and the last column is the leak column of the machine data collected at the same time instant.

Table 3.1 Response of Sensors to Leaks

Time	**PS1**	**AS1**	**PS2**	**AS2**	**Machine data**
	(V)	(V/s)	(V)	(V/s)	**(Leak) (no unit)**
9:49:00 PM	6.78	35.59	5.67	19.63	0
9:49:02 PM	6.02	115.08	8.22	99.00	0
9:49:04 PM	7.38	46.83	5.65	19.88	0
9:49:06 PM	7.06	33.73	5.89	20.14	0
9:49:08 PM	6.46	55.64	6.46	29.17	0
9:49:10 PM	7.02	62.06	6.17	40.80	0
9:49:12 PM	5.92	43.82	5.51	30.94	0
9:49:14 PM	6.60	36.46	5.27	19.16	0
9:49:16 PM	5.99	34.94	5.65	19.40	0
9:49:18 PM	6.91	70.49	7.40	43.13	0
9:49:20 PM	6.49	35.32	5.53	19.58	0
9:49:22 PM	6.68	93.51	6.43	109.91	0
9:49:24 PM	6.18	41.03	6.39	52.50	0
9:49:26 PM	6.56	57.90	5.69	19.37	0
9:49:28 PM	6.07	45.29	5.86	20.04	0
9:49:30 PM	6.98	41.00	5.97	19.93	0
9:49:32 PM	6.04	37.12	5.59	19.17	0
9:49:34 PM	6.79	43.73	6.08	20.40	0
9:49:36 PM	5.91	73.41	6.89	46.44	0
9:49:38 PM	6.91	38.25	5.76	19.96	0
9:49:40 PM	6.96	62.39	6.81	24.99	0.64
9:49:42 PM	5.73	117.05	8.01	110.77	0.86
9:49:44 PM	4.43	131.69	5.06	231.28	0.94
9:49:46 PM	3.48	148.87	7.82	179.35	1.16

9:49:48 PM	3.32	170.76	9.34	165.11	1.38
9:49:50 PM	3.82	176.10	8.84	163.35	1.38
9:49:52 PM	3.86	171.92	8.49	161.30	1.38
9:49:54 PM	3.64	169.64	8.23	152.59	1.38
9:49:56 PM	3.47	168.71	7.88	144.85	1.38
9:49:58 PM	3.42	168.64	8.37	144.91	1.38
9:50:00 PM	3.34	168.42	8.49	167.28	1.38
9:50:02 PM	3.86	165.11	7.90	157.34	1.38
9:50:04 PM	3.91	166.29	7.96	129.82	1.38
9:50:06 PM	3.95	158.19	8.06	153.95	1.38
9:50:08 PM	5.08	130.47	7.67	112.29	0.82
9:50:10 PM	6.19	44.72	5.56	29.59	0.46

It is seen that, in normal conditions i.e. when there is no leak and the leak column of the machine data holds a value of zero, the pressure output PS1 and PS2 corresponds to the set pressure and AS1 and AS2 stays in the range of 0 – 100. But when the leak is created and the machine data leak column value increases, the value of PS1 decreases below 4 and the value of PS2 increases above 5. Also the value of AS1 and AS2 increases to a value greater than 100. Similar tests were performed repeatedly based on which the thresholds are set for each sensor. The software is programmed in such a way that the time instants at which the PS1 value becomes less than 4, PS2 becomes greater than 5 and the values of AS1 and AS2 becomes greater than 100, the value of the mask leak becomes 1. It stays at zero at all other time. As there are some noises in the signals, when the pressure is set at 4, the PS1 signal might give a value less than 4. So, the software is programmed in such a way that if the audio sensors' values are normal, then the PS1 value will always be greater than or equal to 4. Even if the voltage corresponds to a lower value than 4 due to noise, the value will be reassigned to 4. Therefore, the value of PS1 goes below 4 only when there is a leak.

Similarly, various test runs were performed by creating a leak on the instrument side. It is found that when a leak occurs on the instrument side, the PAP machine automatically turns off after few seconds. Thus all the 4 sensors' output drops to a very low value. After the machine stops, the PS1 and PS2 values stays less than 1 and the values of AS1 and AS2 goes below 50. But the few seconds after the leak starts and before the machine stops, the value of AS2 goes above 150. As the machine stops when there is a leak on the instrument side, the machine does not record data and therefore, it was not possible to compare with

the machine data. But the sensors' response during this time was noted and based on the response the threshold conditions were given.

After these conditions are determined, additional two columns, Mask leak and machine leak are added to the data that gets exported from the software. They hold values 0 and 1 indicating absence and presence of leaks at their corresponding location. Leaks were created purposefully for a certain period of time and the data were recorded. A sample data collected is given in below figure (Figure 3.5). The column G represents the leak indicated in the PAP machine data, column H indicates the mask leak and column I named 'Instrument leak', represents the leak on the machine side indicated in the system data. The columns are rearranged to make it easy for comparing. The comparison between the leak column of the PAP machine data and the mask leak column of the data exported from the software is done to check the accuracy of the leak detector system. The green coloured cells represent the times at which leak was created.

Also in another trial, the machine and the software was run for 2 minutes and leaks were created. The mask leak was created from 30th second to 60th second and the machine leak was created from 90th second till the end. The data was recorded and plotted using MATLAB to visualize the leak indication. The x axis represents the differential time from the start time in unit of seconds. The y axis represents the leaks. The mask side leaks are shown in Figure 3.6 and the machine side leaks are shown in Figure 3.7. In the figure, the unit of PS1, PS2 and PS1 – PS2 is V, the unit of AS1 and AS2 is V/s and the unit of time is s. There is no unit for mask leak and machine leak.

	A	B	C	D	E	F	G	H	I
1	time	PS1	AS1	PS2	AS2	ps1-ps2	Machine Leak	Mask Leak	Instrument leak
2	0	0	0	0	0	0	0.04	0	0
3	2	4.472505	105.8437	4.892104	123.0022	-0.4196	0.04	0	0
4	4	5.010054	125.8719	5.406368	142.1118	-0.39631	0.04	0	0
5	6	4.445385	103.1675	4.796283	124.0402	-0.3509	0.04	0	0
6	8	4.876072	130.7972	5.150615	144.0737	-0.27454	0.04	0	0
7	10	4.132894	99.2663	4.540185	117.0196	-0.40729	0.04	0	0
8	12	4.709707	115.6471	4.876938	133.0772	-0.16723	0.04	0	0
9	14	4.565605	102.5725	4.549147	121.292	0.016458	0.04	0	0
10	16	4.422312	126.6323	5.051692	142.7406	-0.62938	0.04	0	0
11	18	4.51015	100.5525	4.841091	121.9748	-0.33094	0.04	0	0
12	20	4.353095	113.7471	5.084092	133.767	-0.731	0.1	0	0
13	22	6.406549	172.7546	9.746591	168.6549	-3.34004	0.66	1	0
14	24	5.051747	172.923	7.237314	169.7544	-2.18557	1.08	1	0
15	26	4.738851	173.6613	7.364846	168.344	-2.626	0.86	1	0
16	28	4.494363	166.53	6.123305	166.3729	-1.62894	0.84	1	0
17	30	4.626727	166.7392	6.298403	169.8177	-1.67168	0.9	1	0
18	32	4.906026	165.1255	6.001632	171.5924	-1.09561	0.88	1	0
19	34	4.570462	165.2647	6.328045	170.5863	-1.75758	0.86	1	0
20	36	4.663562	160.3392	6.028517	167.0909	-1.36496	0.86	1	0
21	38	4.798759	160.4377	6.24153	166.3837	-1.44277	0.82	1	0
22	40	5.12825	141.8704	5.666258	152.773	-0.53801	0.62	0	0
23	42	4.577748	110.6876	4.968968	131.0601	-0.39122	0.06	0	0
24	44	4.261209	116.065	4.325793	135.4625	-0.06458	0.3	0	0
25	46	4.519055	134.6646	5.330194	152.2223	-0.81114	0.02	0	0
26	48	4.543342	101.5703	4.587062	121.7822	-0.04372	0.16	0	0
27	50	4.68542	131.7088	5.523215	154.6079	-0.8378	0.08	0	0
28	52	4.444575	103.8415	4.866943	125.2069	-0.42237	0.1	0	0
29	54	4.82345	116.6369	5.156475	135.9293	-0.33302	0.08	0	0
30	56	4.730351	97.62353	4.952079	119.523	-0.22173	0.06	0	0
31	58	4.761114	108.3586	4.987581	130.1324	-0.22647	0.06	0	0
32	60	4.280234	105.916	4.828338	123.7747	-0.5481	0.08	0	0
33	62	4.04708	123.9599	4.997577	143.3195	-0.9505	0.16	0	0

Figure 3.5 Sample data

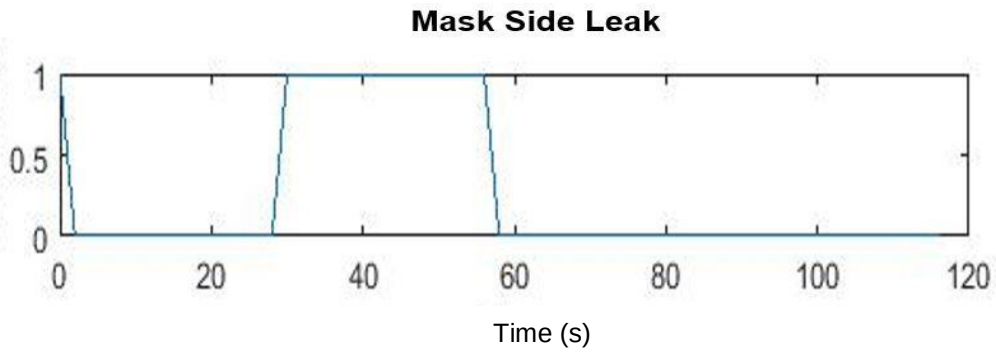


Figure 3.6 Mask leak vs time

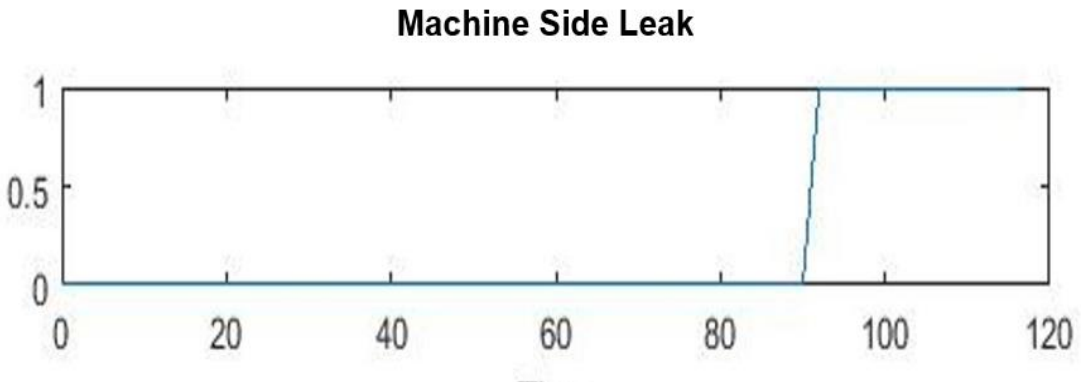


Figure 3.7 Machine side leak vs time

Thus the leaks are indicated based on the thresholds and conditions given. All the four sensors' reactions are checked for mask side leak as well as for the machine side leak. When the leak is on the mask side, the value of PS1 decreases, PS2 increases and the values of AS1 and AS2 increases. When the leak is on the machine side, the values of all the sensor decreases. Whenever a leak occurs, the machine always tries to compensate the leak by increasing the pressure. This is the reason for the increase in the PS2 value during mask leak. As there is a leak on the mask side and air comes out on the mask side, the pressure on the mask side does not increase. The value of PS1 decreases.

3.4. Messaging Service

After detecting the leaks, the next step is to notify the user about the leaks. For this, SMTP, A Simple Mail Transfer Protocol is used. This is the protocol used for e-mail transfers. This can also be used to send text messages to a mobile number when the carrier is known. To do this, the software would need an E-mail account and authentication has to be provided for the software to use the account to send the text message to the user. The information required from the user is the 10-digit mobile number and the carrier they use like AT & T, T-mobile, Verizon etc. Currently this facility is available only for network inside the United States. The start time of the leak is noted and the differential time is calculated for a continuous leak. If the differential time crosses the period mentioned by the clinician, then the message will be sent.

3.5. Disadvantages

The algorithm has some errors in it. Since the threshold for pressure sensor is directly set to 4, not every time the pressure drops below 4 during leaks. When the pressure is higher, for example, if the pressure is higher than 8 and when there is a leak, it might not drop below 4. This is also because the machine tries to compensate for the leaks and does not allow the pressure to drop to a greater extent. Therefore, this algorithm has to be altered to detect the leaks accurately.

4. Version 2

As mentioned in the previous section, the algorithm has to be altered in order to detect the leaks accurately. Also, the output range of the pressure sensor for the pressure of 4-20 cm of water was very low. So, before adjusting the algorithm, thoughts were given in replacing the sensors. Again a pressure sensor and an audio sensor was chosen by evaluating various sensors.

4.1. Pressure Sensor

Again the pressure sensors are evaluated and MPXV5004GC7U was found to be an optimal one. The sensor is manufactured by Freescale Semiconductor. It is a differential and gauge pressure sensor. The top view of pinout of the pressure sensor is given in the Figure 4.1 below. The Sensor that was used previously was designed for a range of 0-70 cm of water. So, the output range for 0-20 cm of water that is required for the PAP machine is very low.

This pressure sensor is designed to measure pressures of range 0-36 cm of water. Also this sensor has an inbuilt amplifier that amplifies the pressure signal to a range of 0 – 5V. The sensor requires a supply of 5V for operation and its output voltage varies in accordance with the change in pressure values as follows,

$$V_{out} = \mathrm{VS} * [(0.2 * \mathrm{P}) + 0.2]$$

Where V_{out} is the output voltage

VS is the supply voltage (5V)

P is the input pressure in units of kPa.

Thus the output voltage range for 4-20 cm of water or 0.39-1.96 kPa is 1.39 to 2.96. The accuracy of the sensor is ± 2.5 % FSS.

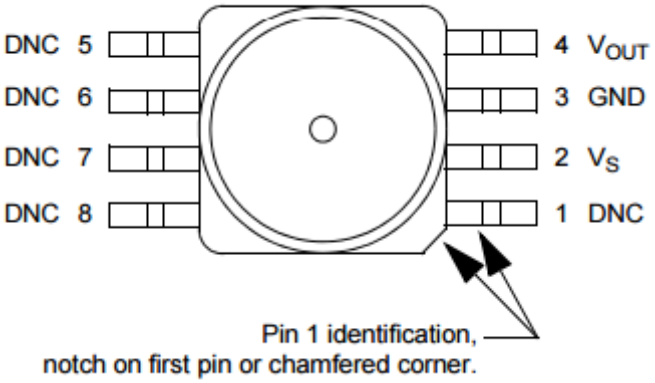
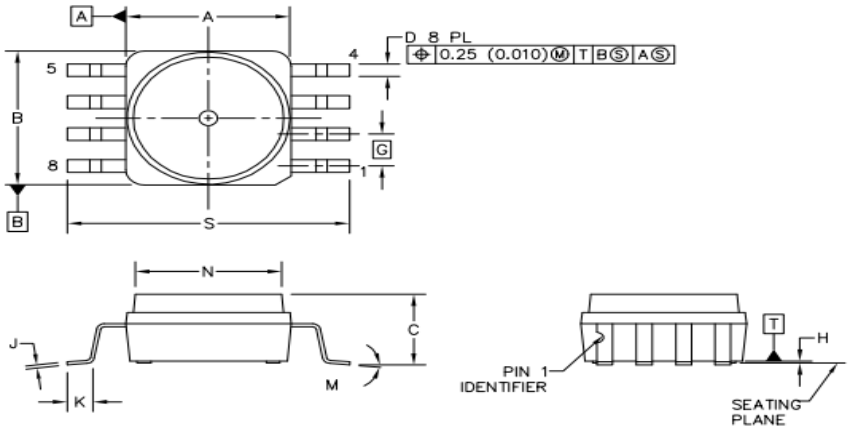


Figure 4.1 Top view of MPXV5004GC7U pinout
Source: MPXV5004GC7U datasheet

The dimensions of the sensor are described below in Figure 4.2



DIM	MILLIMETERS MIN	MAX	INCHES MIN	MAX
A	10.54	10.79	0.415	0.425
B	10.54	10.79	0.415	0.425
C	5.38	5.84	0.212	0.230
D	0.96	1.07	0.038	0.042
G	2.54 BSC		0.100 BSC	
H	0.05	0.25	0.002	0.010
J	0.23	0.28	0.009	0.011
K	1.55	1.80	0.061	0.071
M	0°	7°	0°	7°
N	10.29	10.54	0.405	0.415
S	18.01	18.41	0.709	0.725

NOTES:
1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M, 1994
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION "A" AND "B" DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.15 (0.006).
5. ALL VERTICAL SURFACES 5° TYPICAL DRAFT.

Figure 4.2 Dimensional drawings of the Pressure Sensor
Source: MPXV5004GC7U datasheet

The pressure sensor is commonly used for microprocessor and microcontroller based systems and respiratory equipment. Figure 4.3 shows the block diagram of the internal circuitry of the IC.

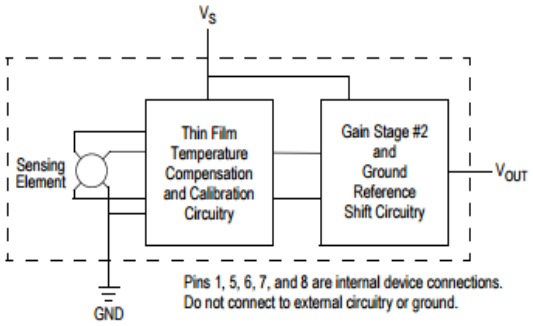


Figure 4.3 Integrated pressure sensor Schematic
Source: MPXV5004GC7U datasheet

The FSS of the pressure sensor is 3 V for 30 cm of water, the offset is typically 1V and the sensitivity of the sensor is 1V/10 cm of water. The output voltage is said to follow a linear relationship with the input pressure. Figure 4.4 shows the relationship between the output voltage in units of volt and input pressure in units of kPa. 1kPa = 10.1972 cm of water.

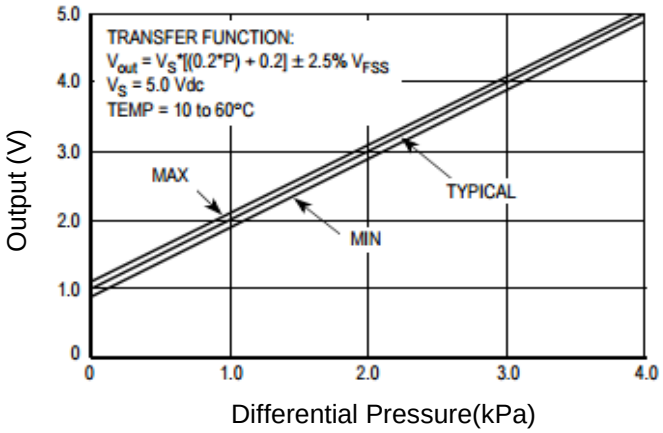


Figure 4.4 Output (V) vs. pressure differential(kPa) at ± 2.5 VFSS

4.2. Audio sensor

Similarly, the audio sensors have also been replaced for better response and performance. Various audio sensors have been again evaluated and Sparkfun Electret Audio sensor BOB-09964 was found to be an

optimal one to use. The component is a Breakout Board (BOB) that uses an electric condenser audio sensor CEM-C9745JAD462P2.54R soldered on to a printed circuit board with an amplifier circuit. The top view of the audio sensor is shown in Figure 4.5 given below.

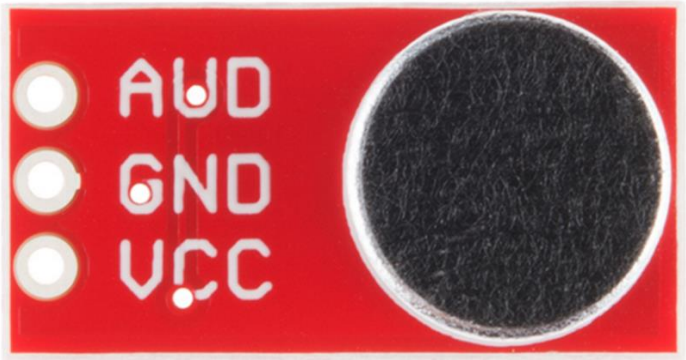


Figure 4.5 Top view of Sparkfun Audio sensor BOB-09964
Source: Sparkfun BOB-09964 datasheet

The audio sensor installed in this break out board is Omni directional, 9.7 mm diameter and 4.5 mm high. The sensitivity of this sensor is -44 dB. This sensor also has two terminals like the audio sensor used previously. One terminal is the output of the sensor and the other terminal is the GND. The other sensor specifications are given in the table **(Table** 4.1) below

Table 4.1 Sensor Specifications

Operating Voltage (V)		**Frequency Range (Hz)**		**Load Resistance (Ω)**	**Temperature (°C)**		**Sensitivity (dB)**	**Sensitivity Reduction**
Min	Max	Min	Max		Min	Max		
1	10	100	10000	680	-20	60	-44	3V to 2V -3dB

The BOB also has an amplifier circuit in it along with the audio sensor, to amplify the sensor voltage to a range of 0-5V. The Breakout board uses a Texas Instruments Op-Amp OPA344 to amplify the signal. The BOB has 3 terminals: VCC, GND and AUD where VCC and GND can be used to power up the device. The rear view of the BOB is given in the figure 4.6 given below.

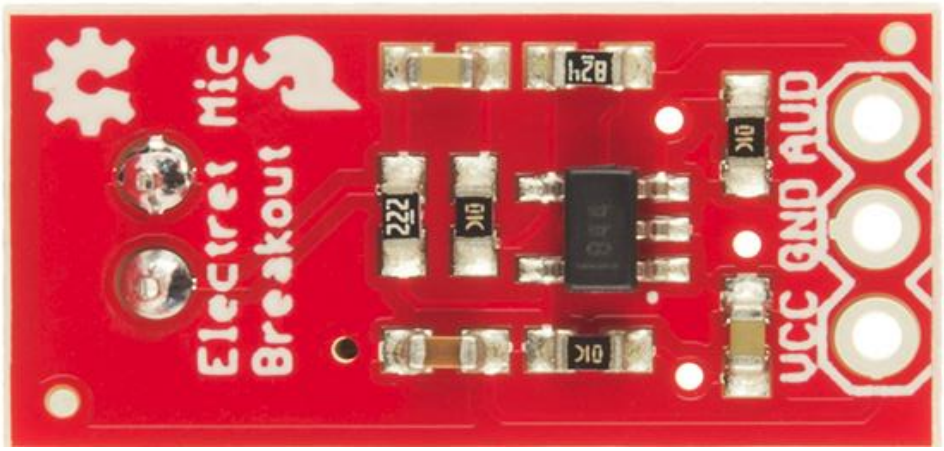


Figure 4.6 Rear view of the Audio Sensor BOB
Source: Sparkfun BOB-09964 datasheet

The signal then goes to the amplifier circuit. The gain of the amplifier circuit is approximately 82V/V. The board was designed by the manufacturer using the Eagle software and the schematic of the circuit is given below in Figure 4.7

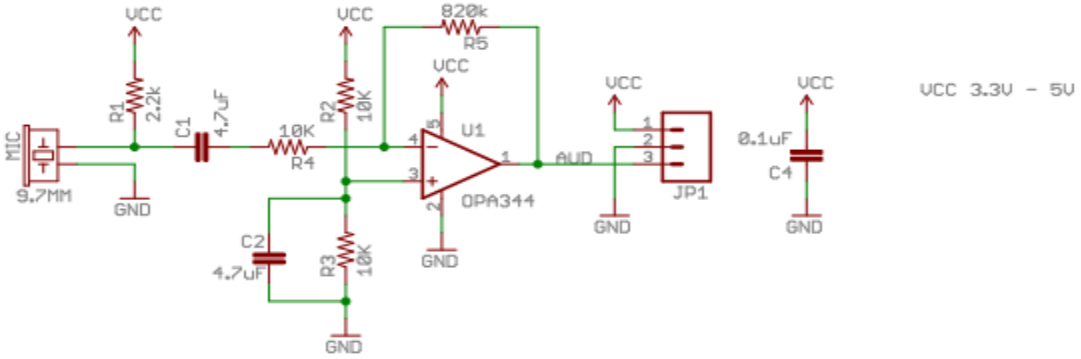


Figure 4.7 Schematic of the BOB

Thus the output voltages of both the sensors are in the range of 0-5V unlike the version 1 sensors whose output voltage is in order of few millivolts. Therefore, in this version, the amplifier circuit can be eliminated completely reducing the number of components and complexity of the system. Therefore, the sensor output is directly connected to the LabJack. The output of mask side pressure sensor is named as PS1 and the output of mask side audio sensor is AS1. Similarly, the machine side sensors are named as PS2 and AS2 respectively.

4.3. Algorithm

The algorithm is adjusted to overcome the issues mentioned in the previous section. The threshold levels are adjusted based on the new sensors. The initial processing is similar to the previous algorithm. Only the leak detection logic is adjusted. Again for this algorithm the sensor data is first collected and analysed by comparing with the machine data. It has been observed that the pressure sensor value dropped or increased to a different value for different pressures. Also, as the pressure increases to a higher value (say 10 cm of water), the audio sensor values were also increasing due to the vibration caused by the air. Thus, it was not possible to detect leaks with one threshold value. Therefore, instead of classifying into two, the sensor values are classified into multiple level and the threshold was set for each level. Based on the combination of the sensor values, the particular level outputs a 0 or 1 for the mask leak and machine leak. It was also found that PS1 and PS2 were giving a different value for different pressures but the difference between PS1 and PS2 (PS1-PS2) was varying depending on the intensity of the leaks. Therefore, instead of using the pressure sensor values separately, the difference is calculated and an additional column called PS2-PS1 was added. The AS1 reacted the same as AS2. Therefore, AS1 was not taken into account for this version and only PS1-PS2 and AS2 were considered for leak classification. A table (Table 4.2) given below indicates the type of leak based on the range of PS1-PS2 and AS2. The AS2 values ranges between 0 – 130.

Table 4.2 Range of PS1-PS2 and AS2 for no leak, mask leak and machine leak

PS1 – PS2(V) / AS2(V/s)	0 – 30	30 – 50	50 – 75	75 – 90	90 – 130
< -4	Mask	Mask	Mask	Mask	Mask
-4 to -0.3	Nil	Nil	Mask	Mask	Mask
-0.3 to 1	Nil	Nil	Mask	Machine	Machine
1 to 5	Machine	Machine	Machine	Machine	Machine

When the software is started, after the initial signal processing, the range in which the signals fall was found out and based on which the leaks were classified as No leaks, Mask side leak and Machine side leak.

4.4. Challenges faced

As previously mentioned in section 3.3, the leak column of the machine data has values ranging from 0 to 2 where 0 indicates no leak and 2 indicates 100 % leak. When the leak column shows a value greater than 1, the leak is considered as gross leak. When the value is between 0.5 and 1, it is considered as fine leaks and when the value is between 0.1 and 0.5, those leaks are considered as very fine leaks. Ideally, 0 indicates no leak condition. But sometimes the machine data shows a value of 0 – 0.1 even in the absence of leak. To check the system's working, it was given to the University of North Texas Health Science Centre (UNTHSC) and was tested with different PAP machine and also with a Sleep Apnea patient. They ran the system for a long time and sent the data collected from the Leak detector system as well as the data collected from their PAP machine.

The algorithm was able to detect leaks corresponding to the leak value of the machine data greater than or equal to 0.5. It was not able to detect the very fine leaks occurring near the mask so again changes have to be made to the algorithm to make it detect the fine leaks. Due to the noises in the system, very fine leaks and absence of leaks were not differentiable. A test run was done and fine leaks were created by moving the mask slightly away from the face and the data were recorded. Table 4.3 given below shows the response of AS2 and PS1-PS2 to the test run. The columns with red font represents the times at which the fine leaks were created. The leak column of the machine data showed a value of range 0 – 0.5 during this run. It is seen from the table that the values do not increase much for the fine leaks and that it is not differentiable from the no leak condition.

Table 4.3 Response of the Sensors to very fine leaks

Time (s)	AS1 (V/s)	AS2 (V/s)	ps1-ps2 (V)	Mask Leak (no unit)	Machine Leak (no unit)
1:45:26 PM	0	0	0	0	0
1:45:28 PM	22.63575	28.60743	-0.04299	0	0
1:45:30 PM	30.25472	36.84253	-0.6104	0	0
1:45:32 PM	17.59257	27.5442	-0.07429	0	0

Table 4.3 *Continued*

1:45:34 PM	18.109	27.23818	-0.10411	0	0
1:45:36 PM	18.02653	26.51873	-0.08585	0	0
1:45:38 PM	17.86374	27.20141	-0.08218	0	0
1:45:40 PM	30.75697	36.50274	-0.48655	0	0
1:45:42 PM	21.50612	27.93776	0.008954	0	0
1:45:44 PM	29.57605	36.82063	-0.50205	0	0
1:45:46 PM	18.08976	26.00834	-0.03075	0	0
1:45:48 PM	17.63415	27.38934	-0.05662	0	0
1:45:50 PM	26.52261	31.30649	-0.12223	0	0
1:45:52 PM	19.33238	28.36347	-0.18232	0	0
1:45:54 PM	34.16871	43.29705	-0.73584	0	0
1:45:56 PM	21.74988	30.62988	-0.18515	0	0
1:45:58 PM	38.50707	43.98745	-0.42817	0	0
1:46:00 PM	35.41275	40.13142	-0.2035	0	0
1:46:02 PM	37.67981	42.5568	-0.33956	0	0
1:46:04 PM	33.78619	42.23448	-0.48194	0	0
1:46:06 PM	26.6869	34.55399	-0.19968	0	0
1:46:08 PM	29.77774	37.72199	-0.49726	0	0
1:46:10 PM	22.02444	29.38558	-0.14059	0	0
1:46:12 PM	21.18343	30.51663	-0.19464	0	0
1:46:14 PM	28.62568	38.0181	-0.52445	0	0
1:46:16 PM	19.36071	28.32294	-0.13312	0	0
1:46:18 PM	21.20119	29.36984	-0.1789	0	0
1:46:20 PM	25.03272	32.48558	-0.2509	0	0
1:46:22 PM	19.23403	26.93787	-0.11126	0	0
1:46:24 PM	30.01851	38.65883	-0.56524	0	0
1:46:26 PM	20.89194	27.80994	-0.02175	0	0
1:46:28 PM	28.77417	33.58926	-0.37203	0	0
1:46:30 PM	20.67203	27.14426	-0.03715	0	0

4.5. Alternate Solution – Version 2.1

Before changing the algorithm, the placement of audio sensor is changed. The audio sensor is placed inside the connector with the diaphragm directly facing the airway. This gave a better response compared to the previous placement. The algorithm was able to detect leaks corresponding to machine leak value greater than or equal to 0.2. The audio sensor placement now and then is shown in Figure 4.9 and 4.8 below.

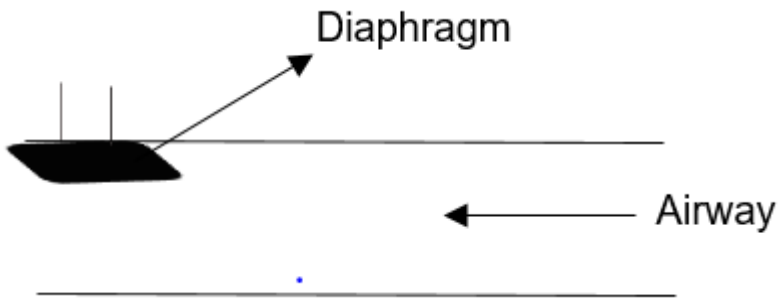


Figure 4.8. Audio sensor placement in previous

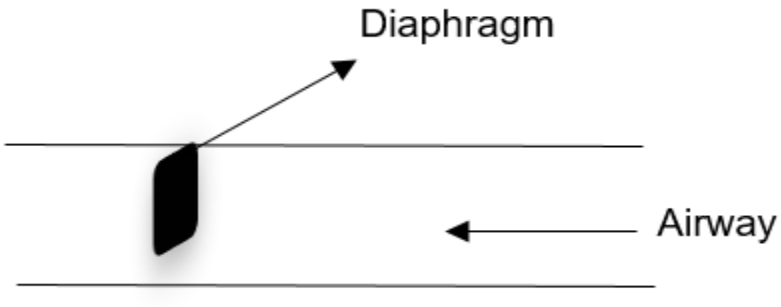


Figure 4.9 Audio sensor placement in version 2.1

But still there were some noises in the system which reduced the accuracy of the system. It was not detecting the leaks all the time. Also the clinician of UNTHSC wanted to detect very fine leaks that occurs in the area of the mask close to the eyes. The leak column of the machine data for those leaks were ranging from 0.02 to 0.2. So changes had to be made to remove the noises and detect the very fine leaks. The table given below (Table 4.4) shows the time, pressure and leak column of the data from the PAP machine for very slight leaks. The 'Time from start' column represents the time difference from the machine start time in units of seconds.

Table 4.4 Machine data for very fine leaks near eyes

Time from Start (s)	MaskPress.2s (cmH_2O)	Leak.2s (L/s)
100	4	0
102	4	0
104	4	0

Table 4.4 *Continued*

106	3.98	0
108	3.98	0
110	3.98	0
112	3.98	0
114	3.98	0.02
116	3.96	0.02
118	3.98	0.04
120	3.98	0.06
122	3.98	0.06
124	3.98	0.06
126	3.98	0.06
128	3.98	0.06
130	3.98	0.08
132	3.98	0.08
134	3.98	0.08
136	3.98	0.06
138	3.98	0.04
140	3.98	0.06
142	3.98	0.06
144	4	0.08
146	3.98	0.08
148	3.98	0.06
150	3.98	0.08
152	4.04	0.08

The system did not respond to these very fine leaks that occurred near to the eyes. To overcome this, the noises has to be filtered completely.

5. Introducing Arduino and MATLAB – Version 3.0

To overcome the challenges and difficulties, to improve the performance of the system and to update the system, a programmable microcontroller is used for acquiring and manipulating the signals. Microcontroller (MuC) is a System on Chip similar to a small computer that has programmable Input/output (I/O) peripherals, memory (RAM and ROM) and a processor. The microcontroller has the capability to read the input from an input device and also write a value to an output device connected to the I/O peripherals. Most commonly used microcontrollers are Arduino, Atmel AVR, Texas Instruments MP4 etc. The MuC also has

several buses for communicating with other devices. The microcontroller chosen for this purpose is Arduino Uno. It is the most efficient board used in design of prototypes. Also to perform the data analysis, a programming software called MATLAB is chosen.

5.1. Arduino – A detailed description

Arduino is an open source platform used in the development of electronic projects. Arduino consists of both hardware and software. The software is also known as Integrated Development Environment (IDE). Programming can be done in the IDE and can be uploaded on to the board and the board can be made to perform any standalone application. Arduino can be used for multiple applications such as to drive motors, LED's sensors and other components. As mentioned in the previous paragraph a microcontroller has memory, I/O peripherals, read write capabilities etc and the Arduino has all the features of the microcontroller [12]. The close up of an Arduino is given in the figure (Figure 5.1) below.

Arduino is of many types and are classified into 4 different categories as follows,

1. Entry Level
2. Enhanced Features
3. Internet of Things (IoT)
4. Wearable

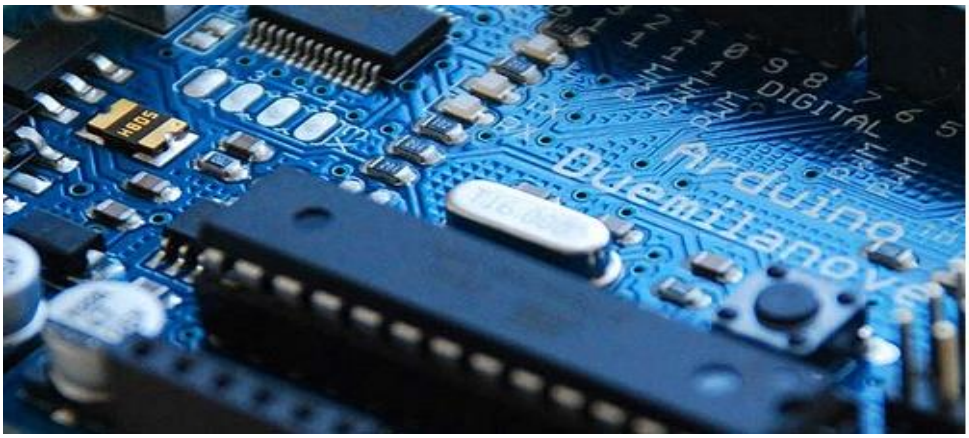


Figure 5.1 Arduino Duemilanove, 2008, (photograph provided by Anthony Mattox) [12]

The Arduino boards in the entry level category are Uno, 101, Pro, Pro Mini, and Micro. The boards in the other level categories are Mega, Zero, Wifi Shield, Gemma etc. Out of which Arduino Uno is the basic board for easy use and for entry level and it is chosen for this project. The schematic of the Arduino UNO board is given in Figure 5.2 given below.

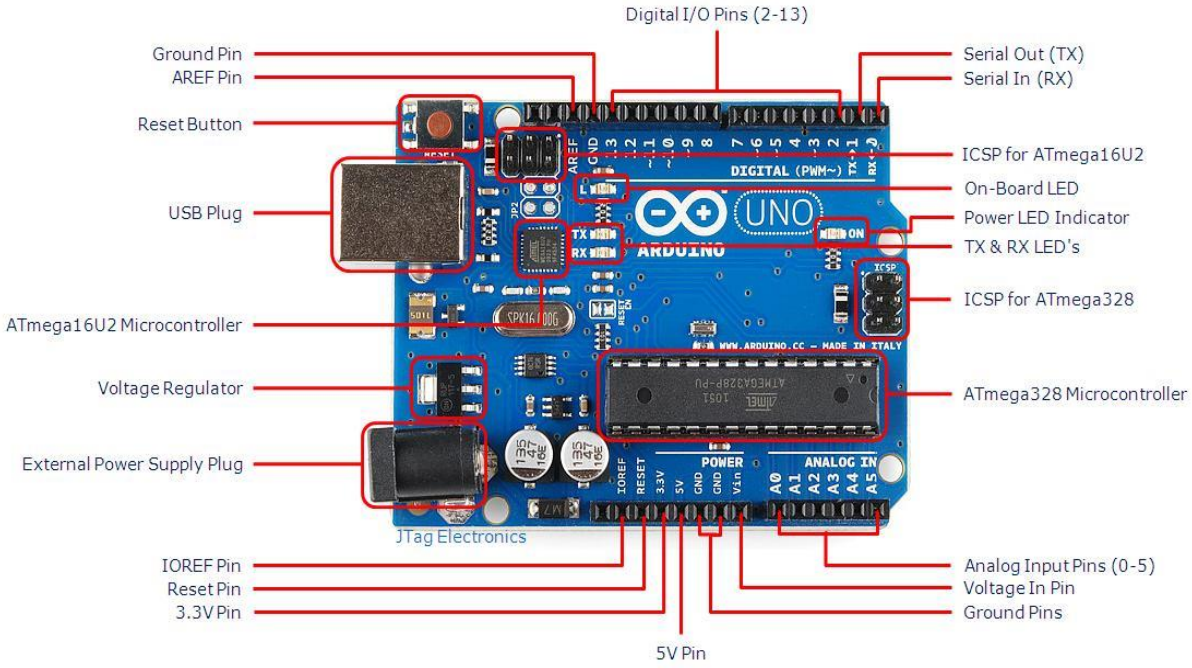


Figure 5.2 Schematic of Arduino Uno board
Source: Arduino Uno datasheet

The Arduino Uno has 14 digital I/O pins of which 6 can be used for pulse width modulation (PWM), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack and a reset button. The board can be powered up by the USB port of a computer or from an external source. The maximum recommended voltage is 12V DC. So either a battery can be used or an AC-DC adapter should be used for connecting to wall-wart. The Arduino Uno has 2 KB of RAM memory and 1 KB of EEPROM memory. Each of the digital pins can be used as input or output. They operate at 5V and can receive maximum current of 40 mA. The board has 6 analog inputs and each connected to different ADC which is capable of providing 10 bits of resolution. The ports are capable of measuring voltages of range 0-5V.

The Arduino has number of communication facilities. They can communicate with Arduino as well as with other microcontrollers. The Arduino uses UART serial communication to communicate through the USB

cable with the computer. The Arduino also has a Software Serial Interface (SSI) that has a master and slaves which can be programmed for communicating with computer as well as wireless device like Bluetooth. The Arduino also has special libraries for I2C communication. For the scope of this project, Serial Communication will be discussed.

The Arduino has a button called ‘reset’ to reset the Arduino to initial state. It also has functions in its library to perform the reset operation by programming. The Arduino Uno also has a resettable polyfuse to protect the computer’s USB ports from shorts and overcurrent. If more than 500 mA is applied to the USB port, the connection will be disconnected by the fuse until the short or overload is removed.

The ports and pins in the Arduino Uno are neatly spaced and organized. The dimensions of the board are described in the figure 5.3 given below

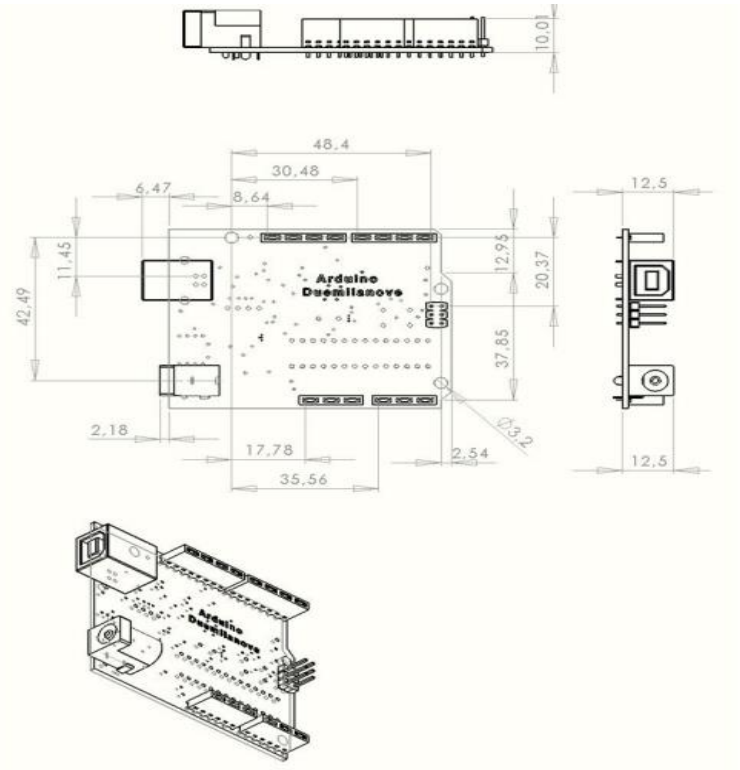


Figure 5.3 Dimensional drawing of Arduino Uno
Source: Arduino Uno datasheet

Based on the dimensions of the Arduino, a box was bought and used to encase the Arduino and its connections. The image of the box is given below in Figure 5.4

Figure 5.4 Arduino Box

One side of the box has a provision for connecting the USB cable (as shown in the left side of Figure 5.4) and other side of the box (shown in the right side of Figure 5.4) has 2 male connectors attached to it. The one that is below the label 'Mask' goes to the mask side sensors and the one that is below the label 'Machine' goes to the Machine side sensors.

5.2. Data acquisition and analysis using Arduino

The analog pins (A0 – A3) are used for acquiring the sensor signals and 5V and GND pins are used to power up the sensors. The connections are as follows,

A0 – AS1

A1 – PS1

A2 – AS2

A3 – PS3

The analog ports in Arduino performs the same task as the analog ports in LabJack. They acquire the analog signals, convert them to digital bytes and send them to the software. Since the ADC used is a 10-bit ADC, the input voltage of range 0 to 5 is converted the digital range of 0 to 2^{10}-1 i.e. 0 to 1023. As

previously mentioned the IDE can be used as a platform for working with the Arduino board. The signals are acquired by defining functions in the software. The two main and mandatory functions are setup() and loop(). The setup function is used for initializing. The loop function is used to read the analog ports and perform operations on the values read. This function is termed as loop because the function gets executed repeatedly until the Arduino is stopped. In this project, the audio sensor signals are preprocessed in the Arduino IDE and its sensitivity is increased before being sent to a programming software.

The Arduino communicates with the programming software using serial port communication. A serial port is a communication channel that transmits one bit of data at a time from transmitter to the receiver at a specific rate called ‘Baud Rate’. There are many serial ports available in a computer known as COM port which will be visible under ‘Device Manager of the computer. The port number and the baud rate can be programmed in the setup function of Arduino, but the same port number and rate should be specified at both transmitter and the receiver end. In this project, the Arduino IDE is the transmitter and a programming software is the receiver.

6. Programming with MATLAB

6.1. Moving Average window

The MATLAB is abbreviated as MATrix LABoratory. As its name specifies, the software stores the data in the form of arrays, vectors or matrices. The MATLAB is programmed to receive the signals from the serial port and store them in an array dynamically. To continuously receive signals a timed loop was built. The loop has a timer that counts time and the tasks inside the loop is executed until the time specified is reached.

As a first step, the acquired signals should be processed to decrease the noise and obtain a meaningful information from the signals. For this purpose, a moving average window of size 500 was designed. This window calculates the average of every 500 samples of data, for e.g. average of data from 1 to 500, from 2 to 501, 3 to 502 and so on. So, the algorithm works only after collecting the first 500 samples. It takes around 20 seconds to collect 500 samples. Thus the moving average filter was designed and a test run was done with a starting pressure of 4 cm of water and then the pressure was increased to 8 cm of water. It was found that the noises have been reduced after calculating the moving average, which is shown in the

figures given below. The Figure 6.1 shows a graph of filtered value of PS1 and the Figure 6.2 is a graph of the filtered value of PS2. In both figures x axis represents the differential time assuming the time instant at which the 500th sample collected to be zero and y axis represents the digital bit value corresponding to the output voltage of the sensor.

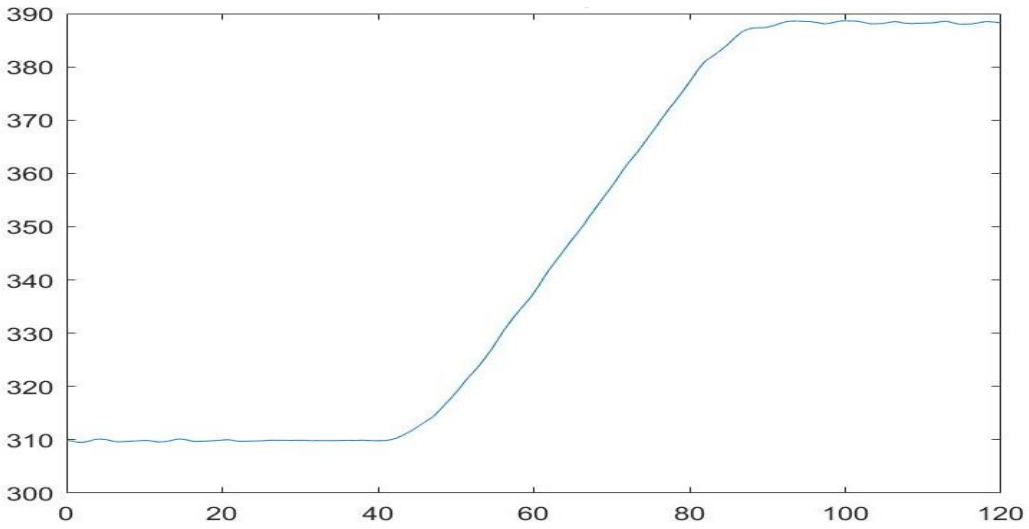


Figure 6.1 filtered PS1 value(bit) vs time(s)

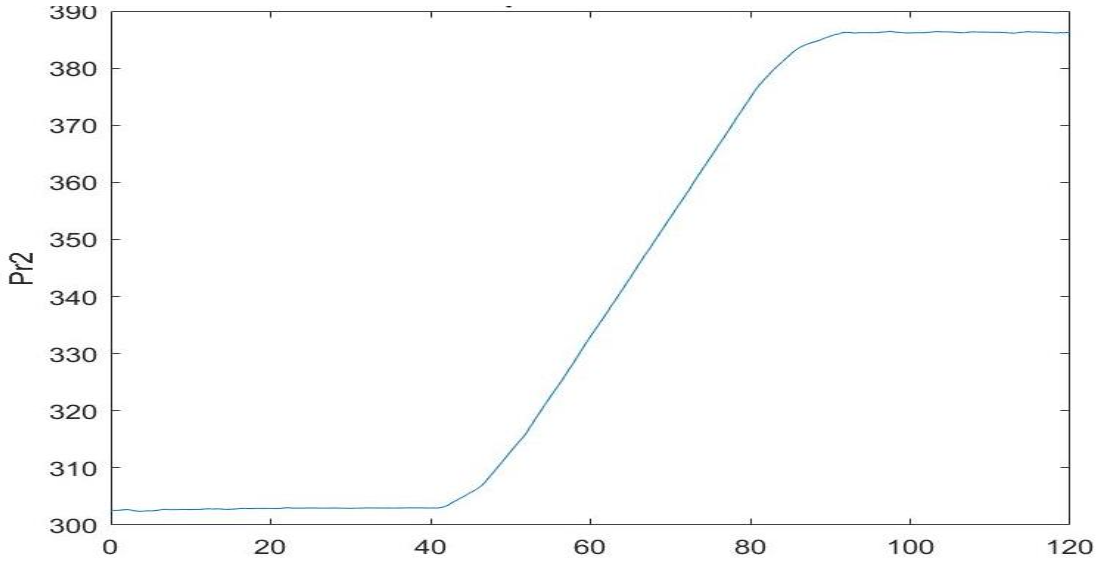


Figure 6.2 filtered PS2 value(bit) vs time(s)

6.2. Moving Average Algorithm

Two different ideas were discussed for detecting the leaks. One is to prompt the user to specify the pressure in the PAP machine at the start (in first 10 seconds) based on which, the sensor values will be calibrated and used for leak detection. But unfortunately, the patients using the PAP machine wouldn't be aware of the pressure being set. Therefore, it is not possible to use this logic. The second is to find the average of the first 15 seconds and use this average for the rest of the time in detecting the leaks. For the first 15 seconds there should not be any leaks in the system. In this case, the algorithm does not depend on the pressure values directly and it does not get affected by the set pressure.

The algorithm works as follows,

1. The averages of all the sensors for the first 15 seconds are calculated. The machine leak and the mask leak for that period is 0.
2. For the rest of the time, the difference between the actual value obtained and the average is calculated. This difference has the nomenclature "d-Sensor name", for e.g. d-PS1 for mask side pressure sensor.
3. It is seen that, when there is a mask leak, the d-PS1 decreases and d-PS2 increases. If this condition occurs the value of mask leak column is changed to 1.
4. When there is a machine leak, both d-PS1 and d-PS2 decreases. Therefore, during this condition, the machine leak column is marked as 1.
5. A baseline was set at -0.2 and 0.2. If d-PS1 and d-PS2 values are in the range -0.2 to 0.2, the mask leak is set to 0.
6. When the pressure increases or decreases manually, the average gets readjusted to the new set pressure value.
7. It is found that the both the audio sensors react the same way for mask leak, machine leak and for manual pressure changes.

   This method is accurate in detecting leaks of all intensities, but it works only for a constant set pressure. But when the pressure is increased manually, both dPS1 and dPS2 increases and

when the pressure decreases, both dPS1 and dPS2 decreases. The sensors respond in the same way as it reacts to the machine leak.

6.3. Version 3.1: Audio Sensor placement change

As mentioned in the previous section, the audio sensors were not helpful in detecting the leaks. It is possible to detect the mask leak without the audio sensors. They are required to detect the machine leaks. Also, while testing, it has been found that an audio sensor placed outside gives a better response than one placed inside. Thus, the mask side audio sensor is eliminated and the location of the machine side audio sensor is changed from inside to outside of the connector. If d-PS1 and d-PS2 decreases and d-AS2 increases and then decreases after the machine stops, then it is machine leak and if d-AS2 doesn't change then it is just manual decrease in pressure. A test run was done for 2 minutes and leaks were created purposefully on the machine side by removing the hose on the machine side. The response of the pressure sensors for machine leaks is shown in the figures (Figure 6.3 and 6.4) given below. The x axis represents the differential time and y axis represents the filtered pressure sensor digital bit values.

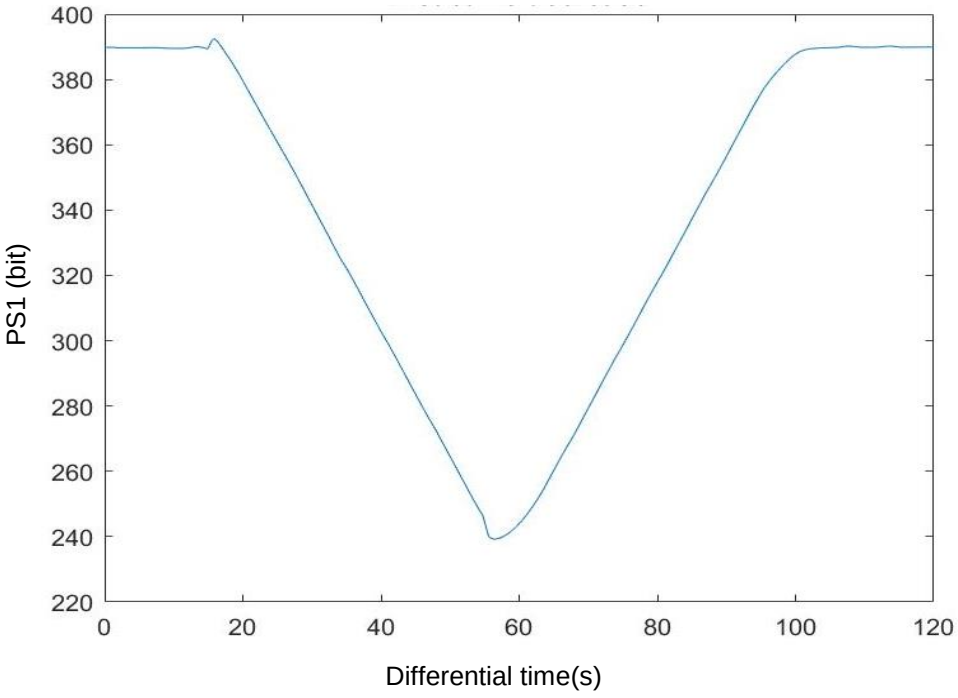


Figure 6.3 Response of PS1 (after filtering) to machine leaks

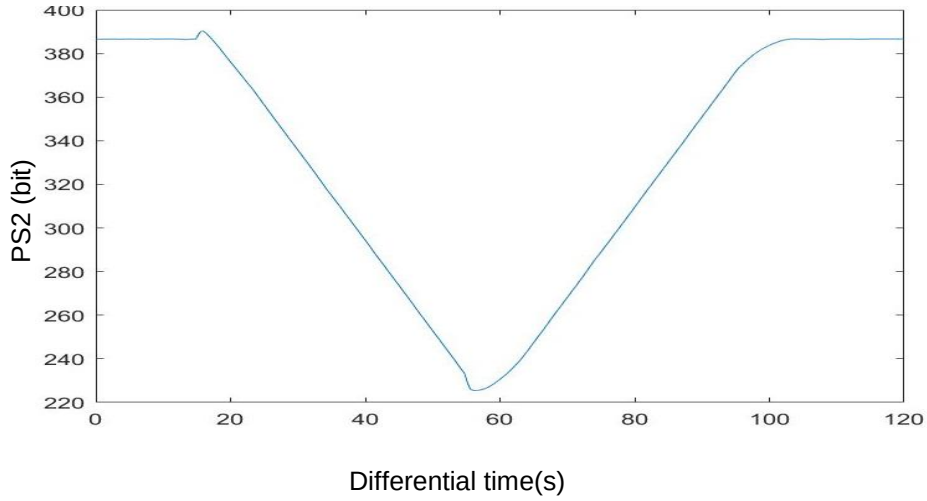


Figure 6.4 Response of PS2 (after filtering) to machine leaks

Similarly, another test run was done by creating very fine leaks on the mask side near the eyes by moving the mask away from the nose. The response of the pressure sensor for mask leaks is shown in the figures (Figure 6.5 and 6.6) given below. The x axis represents the differential time and y axis represents the filtered pressure sensor digital bytes.

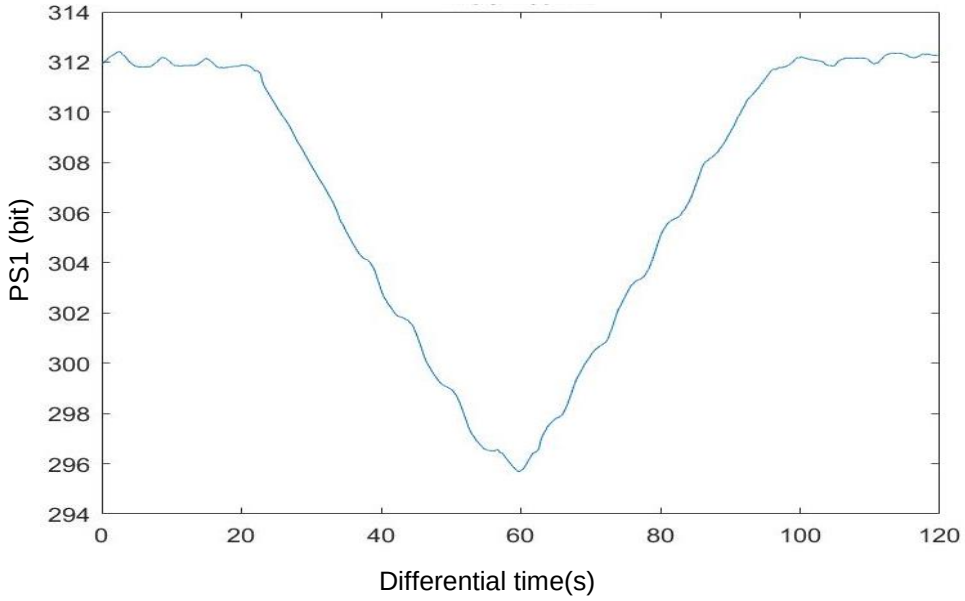


Figure 6.5 Response of PS1 to mask leak

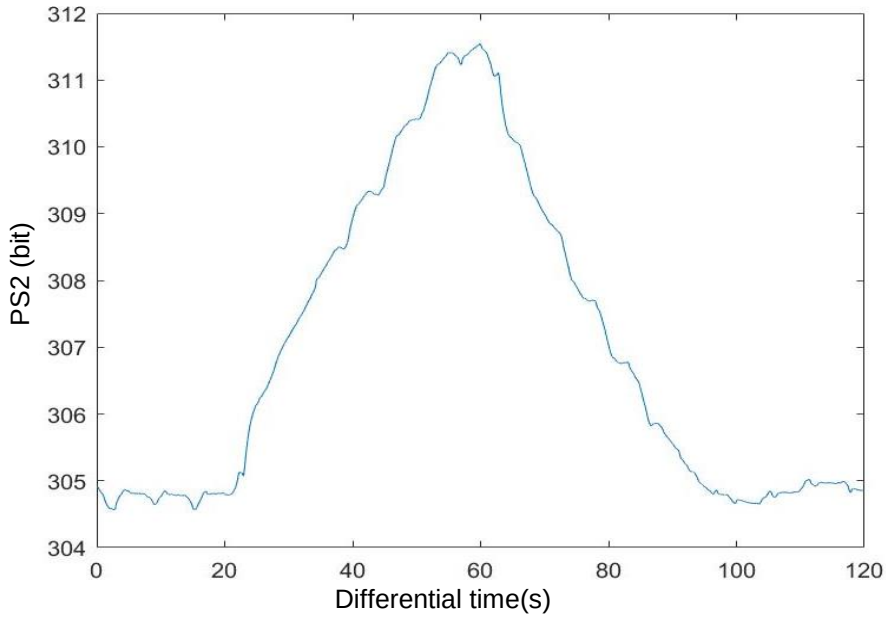


Figure 6.6 Response of PS2 to mask leak

The test run shown in both figures (Figure 6.5 and Figure 6.6) were done for a duration of 2 minutes and leaks were created in between and the responses were noted.

The disadvantage of this algorithm is that it is not possible to maintain a 100% no leak condition for 15 seconds. If a leak occurs accidentally in the first 15 seconds, the average will change and the leaks will not be detected accurately. Another issue is the recalculation of average after the pressure is increased or decreased manually. It takes few seconds to recalculate the average and any leak occurring in between, changes the average and does not detect the leaks accurately.

6.4. Version 4 – Final version

Thus to avoid the averaging issues the algorithm is updated. The PS1 and PS2 values are converted to 4 to 20 pressure range and are again manipulated to minimize the error in the signal such that the PS1 and PS2 values stays close to the pressure being set. The difference between PS1 and PS2 is used for determining the leaks.

6.4.1. Two level filtering Algorithm

For test purpose, again the timed loop was created to acquire and process the data continuously. In this algorithm, instead of manually connecting to the serial port for communication between MATLAB and Arduino, an Arduino MATLAB package was downloaded and installed. The commands in the package allows to detect the Arduino hardware, connect to it and read voltages from Arduino ports. The package also uses the serial port communication but does not require to set up the serial port manually and manage the baud rates.

When the signals are read each time the loop runs, they get stacked into an array and again the moving average is calculated as done in the previous version. The window size is chosen to be 500. Here for the first 500 samples, the window size gets adjusted based on the number of samples acquired. For e.g. If the number of samples acquired is 100, the average of the 100 samples is found. However, the first 30 seconds after the start is allocated for the sensors and averaged values to settle up. The mask and machine leak columns outputs a value of 0 in this time period to avoid false leak detection. Thus it is recommended to run the system for more than a minute in order to detect leaks.

Here, the moving average of the filtered pressure sensor values are again found in order to minimize the noises and improve the accuracy of the system. These 2^{nd} level filtered values are used for leak detection. Since the 2^{nd} level averaging is done the filtered PS1 and PS2 values stays almost close to the set pressure value in absence of leaks. So the PS2-PS1 values shows a clear difference between no leak and very fine mask leak conditions.

A fine leak of order 0.3 L/s was created by moving the mask area away from the chin and the response of 2^{nd} level filtered PS1 and PS2 were noted and the plots of the filtered PS1 and PS2 responses for leaks over time is shown in the figures (Figure 6.7 and 6.8) given below. It is seen that a smoother curve was obtained compared to the other versions of the software.

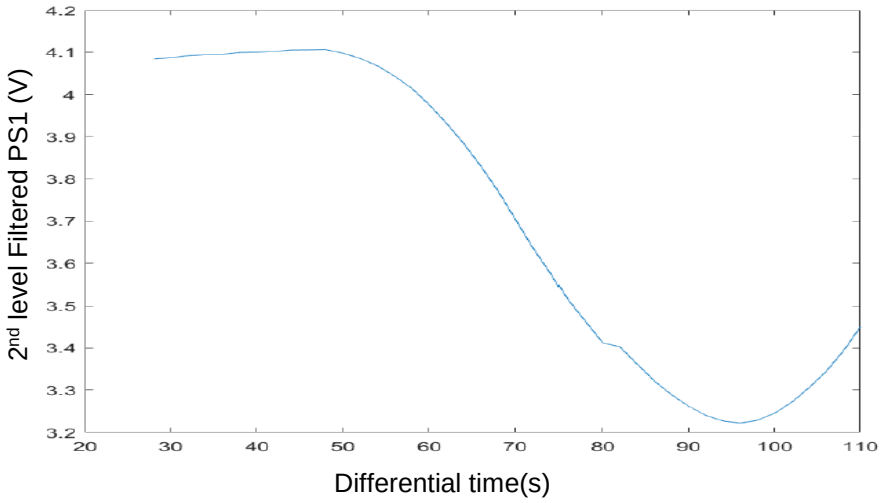


Figure 6.7 Response of 2nd level filtered PS1 to fine mask leak

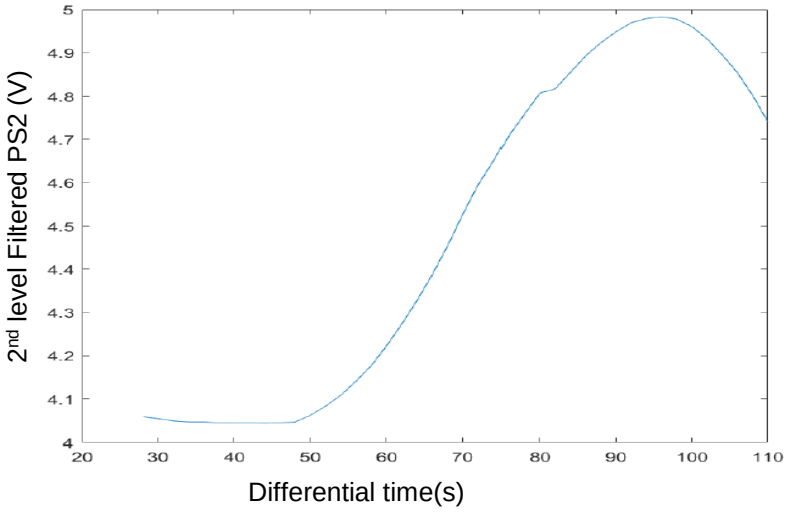


Figure 6.8 Response of 2nd level filtered PS2 to fine mask leaks

The mask leak was created after the 55th second. It can be seen that until the 55th second both PS1 and PS2 stays almost at the same value and when the leak is created, the PS1 decreases and PS2 increases. Therefore, the PS2-PS1 will stay at around 0 in the absence of leaks and increases when the leak occurs on the mask side. The change in the PS2-PS1 is also clearly seen after the 2nd level averaging. The PS1, PS2 and PS2-PS1 data were recorded for the above given test and is shown in the below table (Table 6.1) to indicate the variation between no leaks and fine leaks.

Table 6.1 Data after 2nd level filtration

Differential Time	PS1	PS2	PS2-PS1
40.00	4.10	4.05	-0.05
42.00	4.10	4.05	-0.06
44.00	4.11	4.04	-0.06
46.00	4.11	4.05	-0.06
48.00	4.11	4.05	-0.06
50.00	4.10	4.06	-0.04
52.00	4.09	4.08	0.00
54.00	4.07	4.11	0.04
56.00	4.04	4.14	0.10
58.00	4.01	4.18	0.16
60.00	3.98	4.22	0.25
62.00	3.93	4.27	0.34
64.00	3.89	4.33	0.44
66.00	3.83	4.39	0.56
68.00	3.77	4.45	0.68
70.00	3.71	4.53	0.82
72.00	3.64	4.59	0.96
74.00	3.58	4.65	1.07
76.00	3.51	4.71	1.20
78.00	3.46	4.76	1.30
80.00	3.41	4.81	1.39
82.00	3.40	4.82	1.41
84.00	3.36	4.85	1.49
86.00	3.32	4.89	1.57
88.00	3.29	4.92	1.63
90.00	3.26	4.95	1.69

The data highlighted in red color are the times when leaks were created. The PS2 – PS1 plot for the same values is shown in the figure (Figure 6.9). It is seen that the difference increases prominently during the leak and are easily differentiable.

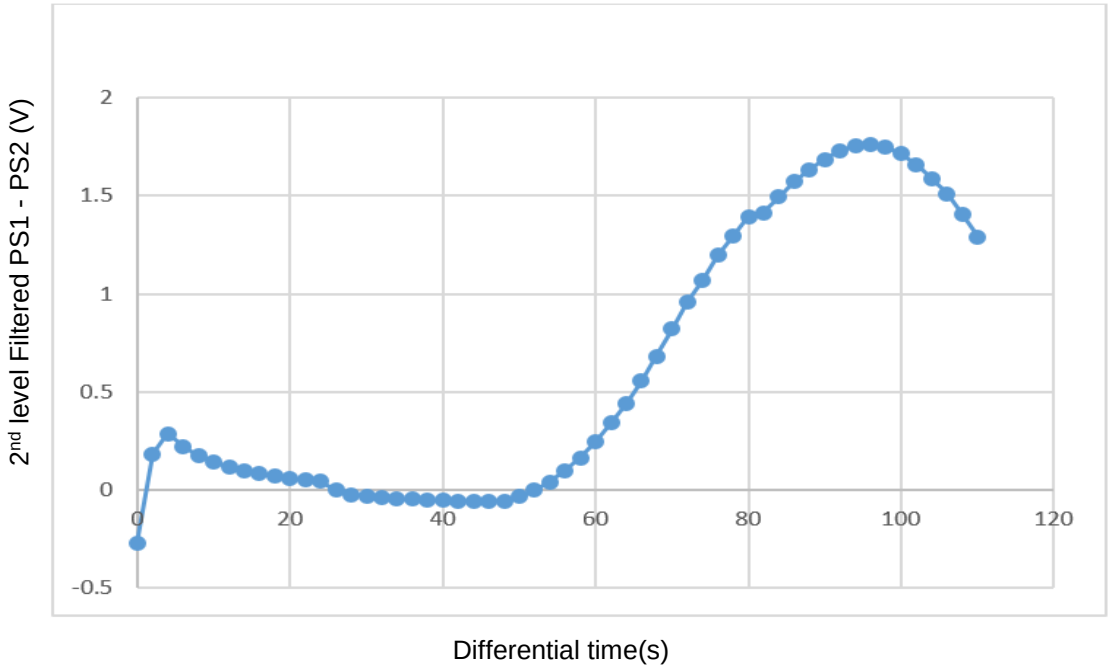


Figure 6.10 2nd level filtered PS2 – PS1 vs Time for mask side leaks

It can be observed from the above plot that the difference is almost 0 when there is no leak and starts increasing when leak occurs. Based on this the threshold was set to 0.3 and whenever the difference is greater than 0.3, the mask leak column is changed from 0. In this version, unlike the mask leak column of the previous versions that holds only two states 0 and 1, the mask leak column here holds values from 0 – 4 depending on the intensity of the leaks where, 0 represents no leak, 1 represents leaks less than 25%, 2 represents leaks from 25-50 %, 3 represents leaks 50-75% leak and 4 represents 75-100 % leak. The percentage was determined based on the machine data. As previously mentioned, the machine data value 0 indicates 0% leak and 2 represents 100% leak.

Similarly, the filtered PS1 and PS2 values were observed for the machine side leaks. Again test run was done and leak was created on the machine side after few seconds. The response of the 2nd level filtered PS1 and PS2 are given in the below figures (Figure 6.10 and 6.11).

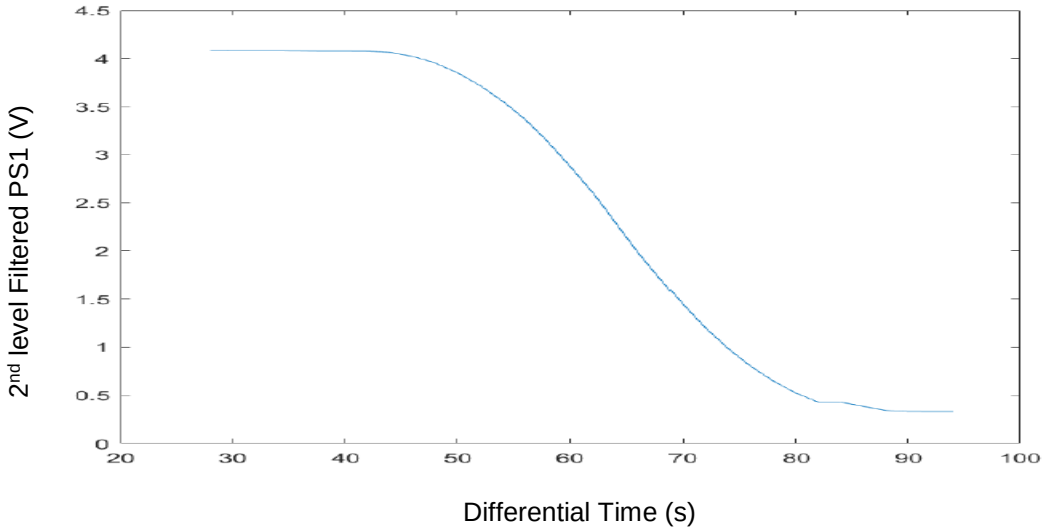


Figure 6.10 2nd level filtered PS1 vs Time for machine side leaks

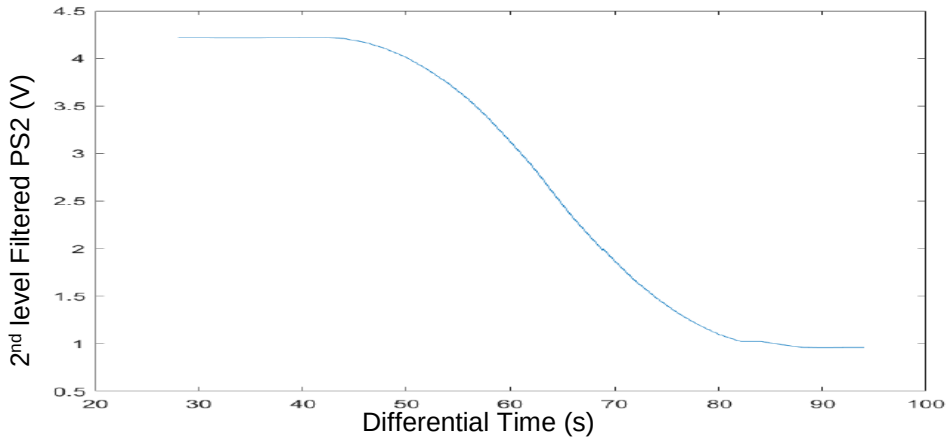


Figure 6.11 2nd level filtered PS2 vs Time for machine side leaks

It can be seen that when there is a leak on the mask side both PS1 and PS2 drops to a low value. So a threshold was set at 2. If both the values drop below 2 then the machine leak value is changed from 0 to 1. For the first 30 seconds the leak values are maintained as 0. The leak detection part of the algorithm starts working after 30 seconds. The algorithm calculates the average and uses it for leak detection. Therefore, when leak occurs and comes back to normal, the averaged sensor values does not come to normal immediately, it takes a certain amount of time to come back to normal. Therefore, after the system stops, the software is programmed to adjust for the delay and the leak values at those times are changed to 0. The software was then tested by creating mask leaks from 45th second to 80th second and the mask leak

values before and after delay adjustment are plotted and shown in below figures (Figure 6.12 and Figure 6.13)

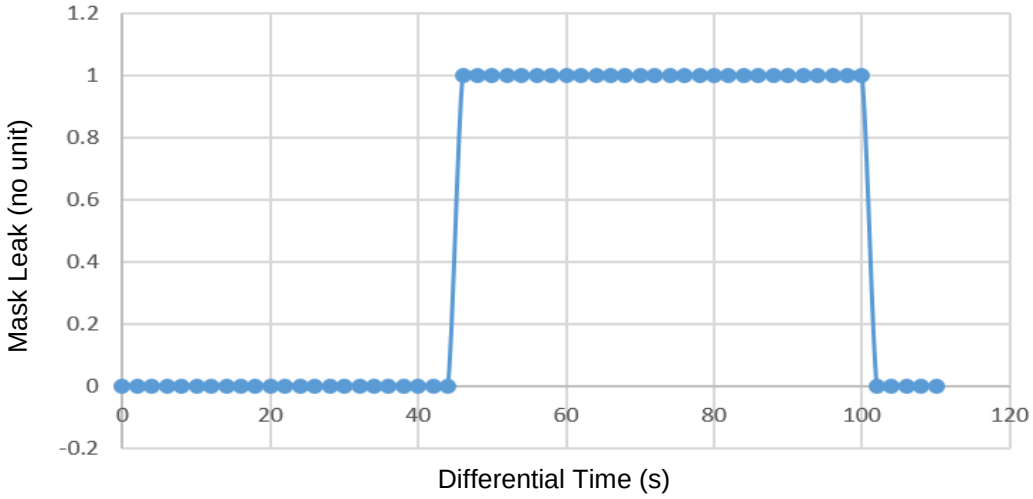


Figure 6.12 Mask Leak vs time before delay adjustment

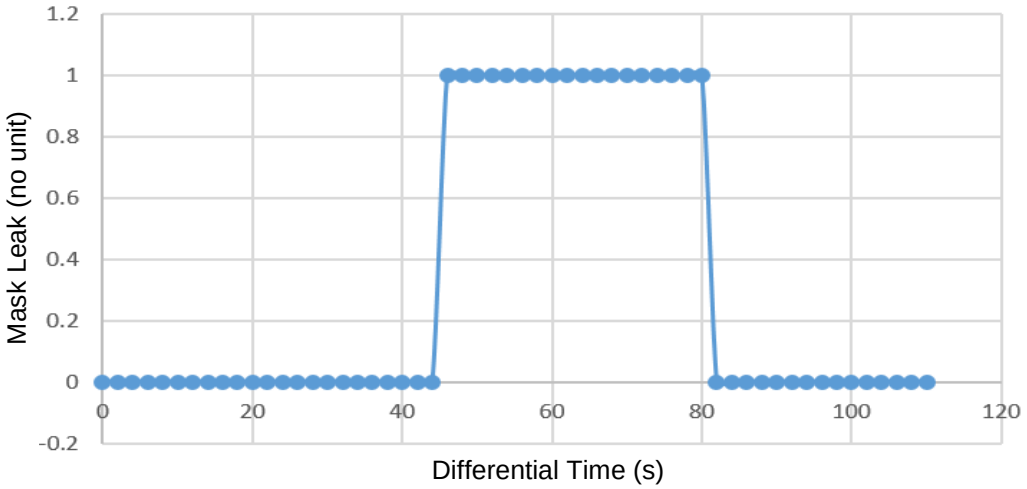


Figure 6.13 Mask Leak vs time after delay adjustment

Thus, it can be seen from the above plots that, before delay adjustment, it took an additional 20 seconds to come back to normal and after delay adjustment, this 20 second delay was adjusted. To record the values every 2s and export it to excel, a timer was designed that counts the time and runs a function every 2s. The function is programmed to store the values of PS1, PS2, PS2-PS1, Mask leak and Machine leak to an array every 2s. After the software is stopped, the values get exported to an excel sheet.

6.5. Notification

Two notification methods were designed. They are,

1. Text Message
2. Alarm

The messaging system is similar to the one explained in section 3.4. MATLAB is programmed to use SMTP protocol to send text messages to a mobile number. Authentication is required and the mobile number and the carrier should be known. The start time of the leak is noted and indicated along with the location and intensity percentage of the leak in case of mask leak. The intensity of the leak is calculated based on the percentage calculation explained in the previous section (section 6.4.1). Based on the value in the mask leak column from 0 to 4, the intensity percentage holds value 0 to 100%. For machine leaks, the time and location of the leak is notified. Another notification method was also designed. It is the alarm system. Whenever the leak stays longer than the specified period, an alarm tone will be played automatically via the computer speakers. Alarm tone will be played for a certain period of time specified by the user.

7. Additional Features

7.1. User Interface

The MATLAB has a run button to start the algorithm but there isn't an option to stop it. Also, as timed loop is used to continuously acquire data, it would be required for the user to specify the duration of the run. It is practically difficult for the user to specify the duration in advance. Therefore, a Graphical User Interface (GUI) was built to allow the user to directly start and stop the program. In MATLAB, functions should be defined for each icon used in the GUI. Initially the GUI was built with 2 push buttons and a toggle button. One issue with the serial port is that it should be disconnected before running the program again. Thus the 2 push buttons are used to connect and disconnect the serial port and the toggle button to start and stop the program. Later, after the manual serial setup was replaced with the MATLAB Arduino package commands, the GUI was updated and left with just 1 toggle button to start and stop the program. The GUI also has a drop down list to with three options as follows,

1. No notification
2. Message
3. Alarm

Based on the option selected from the drop down box by the user, corresponding actions will be taken. If the user chooses ‘No notification’, the user wouldn’t be notified of the leaks. The image of the User Interface is shown below in figure 7.1

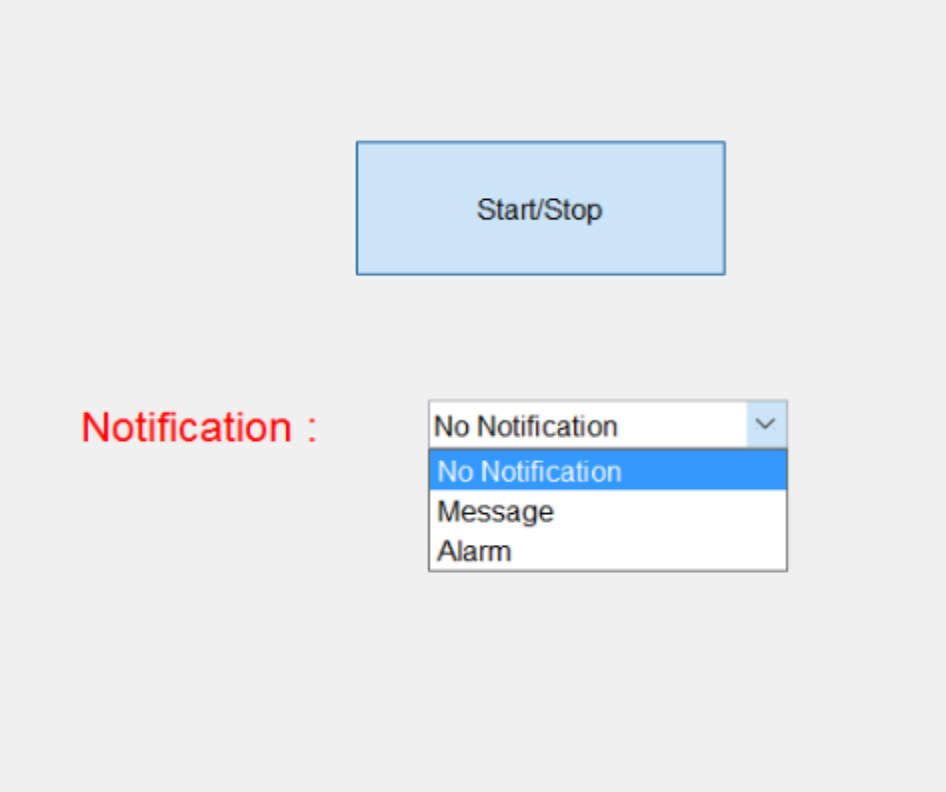


Figure 7.1 User Interface Screen

The Start/Stop button pressed once starts the software and pressed again, stops the software. When the start button is pressed, a dialog box as shown in Figure 7.2 will appear that prompts the user to enter the date and time which will be used to name the data file that gets exported.

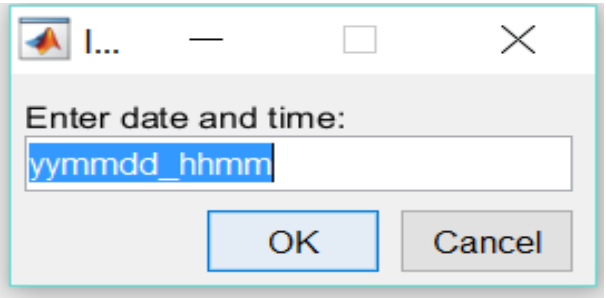


Figure 7.2 Date and Time dialog box

The algorithm starts only after the date and time is entered and the 'OK' button is pressed.

7.2. Report Generation

The software is programmed to generate a report about the leaks intensity and duration during a run after the stop button is pressed. In the beginning, the date and time at which the software started is displayed. Then, the overall percentage of presence of mask leak and machine leaks over time was calculated. This is not the intensity percentage of the leaks. This is the percentage of time the leaks were present over the entire run. Then the plots of the 2nd level filtered PS1 and PS2 were displayed and the information about all the mask and machine leaks were displayed. The start time and end time of the leaks are displayed. For mask leaks, along with the start and end time, the intensity percentage is also displayed. The mask and machine leaks plots are also displayed. A sample report generated is shown in Appendix C.

8. Future work

8.1. Additional Features for UI

The User Interface can be improvised in the future and additional features can be added. The system uses only button to start and stop the system. It doesn't show any difference between the button not pressed condition and pressing the button once for start and pressing again for stop. This may confuse the user if they have started the software or stopped the software. Therefore, an indication can be given to the user to differentiate these conditions. Also the contact number is entered in the code directly for the message notification system. Another option can be included in the user interface to prompt the user to enter the contact number to which the notification has to be sent.

8.2. Wireless System

The additional feature for the system is to remove the wires and turn the system into a wireless one. Many background research was done to choose the appropriate wireless module for the system and have decided to use a Bluetooth module to start as it is simpler and uses the serial port communication to transmit and receive the data. The Sparkfun BlueSMiRF silver Bluetooth module is chosen. It is capable of both transmitting and receiving the data. The front view of the Bluetooth module is shown in the figure given below (Figure 8.1)

Figure 8.1 BlueSMiRF silver Bluetooth Module

The Bluetooth module requires an input voltage of 5V. It has 2 pins Rx for receiving the data and Tx for transmitting the data. The supply for the module is obtained by connecting the supply pin of Bluetooth module to 5V pin of Arduino. The Arduino has to be programmed to transmit the data to the Bluetooth module using the serial communication and the Bluetooth module transmits it wirelessly. This serial communication is different from the serial port communication that was explained previously in section 6.1. The one explained before was hardware serial communication that uses the USB port. For the wireless communication, the Arduino has to be programmed to use the Software Serial Interface for transmitting the data to the Bluetooth module. Any two of the digital pins can be chosen and programmed to act as transmit and receive pins for the software serial port of the Arduino. The Rx pin of the Bluetooth module should be connected to Transmit pin of the Arduino Software Serial port. The Bluetooth receiver in the computer will receive the data. The data obtained can then processed in the same way the signals were processed and analyzed in version 4. The Arduino Bluetooth connection is given in Figure 8.2 given below. The sensors will be connected to the Arduino as done previously and the Arduino will be interfaced with the Bluetooth transmitter and can be programmed to transmit the sensor data. A Bluetooth receiver will be connected to the receiver end which can receive the data. The MATLAB acquires the data through the serial port.

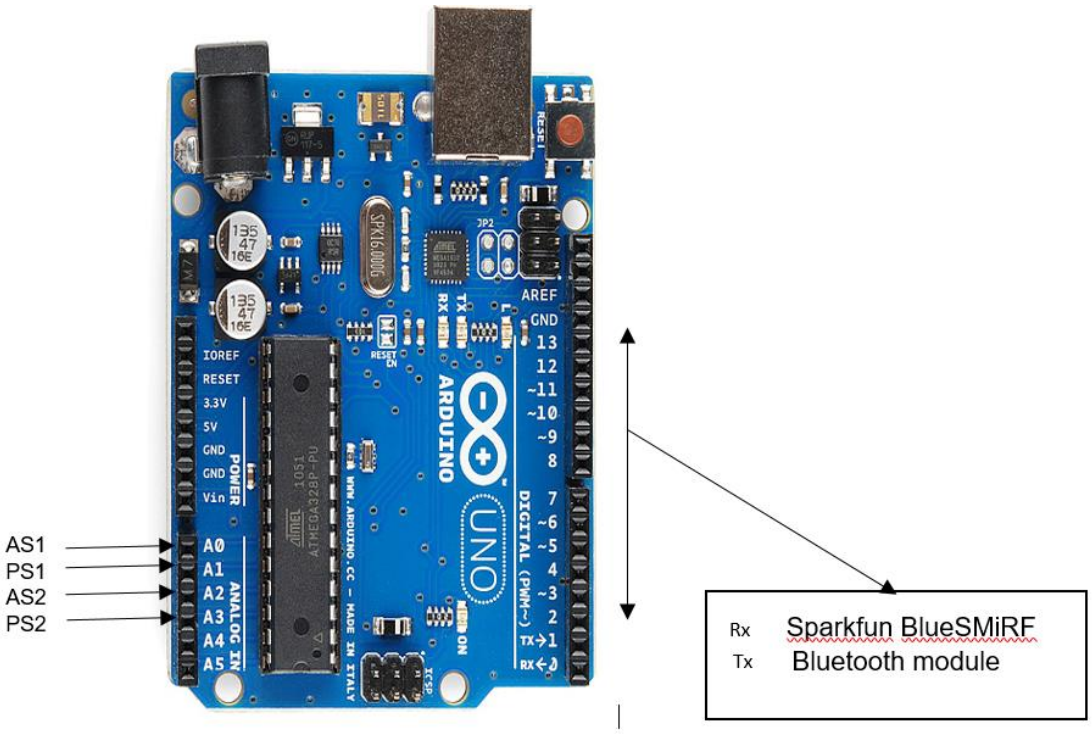


Figure 8.2 Arduino Bluetooth Connection

9. Results and Conclusion

In the first version developed, the system was only able to detect leaks when the set pressure is in the range of 4 – 8 cm of water and was giving erroneous values for higher pressures. In the next version, the Leak detector system was able to detect leaks corresponding to the machine data leak value greater than 0.5. Thus the accuracy of this version was 75%. Then in the next version of the algorithm, the system accuracy was increased to 90%. In the final version of the algorithm, the system was able to detect leaks of all intensities and therefore, the accuracy of the system having version 4 software is 100%. But still additional features can be added in future to improvise the system capabilities.

Thus the two level filtering with pressure difference algorithm works best. It detects all intensities of leaks on mask side and also detects leaks on machine side. The system also has the capability to indicate the time, location and intensity of the leaks whenever the leak remains active for a specific period.

Appendix A

Leak Detection Software

```
% Main code for building User Interface, Data Acquisition and Data Analysis


function varargout = interface(varargin)
% INTERFACE MATLAB code for interface.fig
%      INTERFACE, by itself, creates a new INTERFACE or raises the existing
%      singleton*.
%
%      H = INTERFACE returns the handle to a new INTERFACE or the handle to
%      the existing singleton*.
%
%      INTERFACE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in INTERFACE.M with the given input arguments.
%
%      INTERFACE('Property','Value',...) creates a new INTERFACE or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before interface_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to interface_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

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

% Last Modified by GUIDE v2.5 13-Jul-2016 22:54:54

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @interface_OpeningFcn, ...
                   'gui_OutputFcn',  @interface_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 interface is made visible.
function interface_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 interface (see VARARGIN)
```

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

% Update handles structure
guidata(hObject, handles);

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


% --- Outputs from this function are returned to the command line.
function varargout = interface_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;

%% LEAK DETECTOR
%% VARIABLES EXPANSION
% rawMask - Mask side Pressure Sensor output (units Volts)
% rawMachine - Instrument side Pressure Sensor output (units Volts)
% maskPressure - mask side Pressure value (unit cm of water)
% machinePressure - machine side Pressure value (unit cm of water)
% Two levels of moving average is done to get a smooth curve
% avg1_Mask - First Moving average of Mask side Pressure value
% avg1_Machine - First Moving average of Machine side Pressure value
% avg2_Mask - Second Moving Average of Mask side Pressure value
% avg2_Machine - Second Moving Average of Machine side Pressure value
% maskLeak - To indicate if there is mask leak or not
% The value of the maskLeak ranges from 1 to 4 to indicate various
% intensities of leaks
% machineOff - Indicate if the machine is turned off
%%

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

% Hint: get(hObject,'Value') returns toggle state of startStop

b = 1;
while(get(hObject,'Value'))
   if(b == 1)
      prompt = {'Enter date and time:'};
      dlg_title = 'Input';
      num_lines = 1;
      defaultans = {'yymmdd_hhmm'};
      answer = inputdlg(prompt,dlg_title,num_lines,defaultans);
      assignin('base','answer',answer);
      docstr = char(answer);
```

```
assignin('base','docstr',docstr);
connectArduino = arduino;
i = 1;
j = 1;
k = 1;
l = 1;
m = 1;
co = 1;
co1 = 1;
coI = 1;
coI1 = 1;

persistent rawMask;
persistent rawMachine;
persistent maskPressure;
persistent machinePressure;
persistent maskLeak;
persistent machineOff;
persistent slopePS1;
persistent slopePS2;
persistent avg1_Mask;
persistent avg1_Machine;
persistent avg2_Mask;
persistent avg2_Machine;
persistent pressureDifference;
persistent leakIndex;
persistent timeOfLeak;
persistent percentPressure;
persistent dataWithDate;
persistent n;
persistent output;
persistent dataFor2s;
persistent sumOfLeaks;
persistent leakTimePercent;
persistent leakIndex1;
persistent timeOfLeak1;
persistent sumOfLeaksI;
persistent leakTimePercentI;
persistent movingAverage2Raw;


if isempty(n)
    rawMask(:) = 0;
    rawMachine(:) = 0;
    maskPressure(:) = 0;
    machinePressure(:) = 0;
    maskLeak(:) = 0;
    machineOff(:) = 0;
    slopePS1(:) = 0;
    slopePS2(:) = 0;
    avg1_Mask(:) = 0;
    avg1_Machine(:) = 0;
    avg2_Mask(:) = 0;
    avg2_Machine(:) = 0;
    pressureDifference(:) = 0;
    dataFor2s(:) = 0;
```

```
        leakIndex(:) = 0;
        timeOfLeak(:) = 0;
        percentPressure(:) = 0;
        dataWithDate = 0;
        output(:) = 0;
        n(:) = 0;
        movingAverage2Raw(:) = 0;
    end
    b = 0;
end
rawMask(i,1) = readVoltage(connectArduino,'A0');
rawMachine(i,1) = readVoltage(connectArduino,'A3');

maskPressure(i,1) = ((rawMask(i,1) - 1.527)/(3.025-1.527))*16 + 4;
machinePressure(i,1) = ((rawMachine(i,1) - 1.583)/(3.225-1.583))*16 + 4;

maskLeak(l,1) = 0;
machineOff(l,1) = 0;
slopePS1(l,1) = 0;
slopePS2(l,1) = 0;

if(i<500)
    mfMask = medfilt1(rawMask((1:i),1));
    movingAverageRaw(j,1) = mean(mfMask);
    mfMachine = medfilt1(rawMachine((1:i),1));
    movingAverageRaw(j,2) = mean(mfMachine);
    avg1_Mask(j,1) = mean(maskPressure(1:i),1);
    avg1_Machine(j,1) = mean(machinePressure(1:i),1);
end

if(i>=500)
    mfMask = medfilt1(rawMask(k:(k-1)+500,1));
    movingAverageRaw(j,1) = mean(mfMask); %------ PS1
    mfMachine = medfilt1(rawMachine(k:(k-1)+500,1));
    movingAverageRaw(j,2) = mean(mfMachine); %------ PS1
    avg1_Mask(j,1) = mean(maskPressure(k:(k-1)+500,1));
    avg1_Machine(j,1) = mean(machinePressure(k:(k-1)+500,1));
    k = k+1;

end
if j < 500
    movingAverage2Raw(l,1) = mean(movingAverageRaw(1:j,1));
    movingAverage2Raw(l,2) = mean(movingAverageRaw(1:j,1));
    avg2_Mask(l,1) = mean(avg1_Mask(1:j),1);
    avg2_Machine(l,1) = mean(avg1_Machine(1:j),1);
    pressureDifference(l,1) = avg2_Machine(l,1) - avg2_Mask(l,1);
    end

if j>=500
    movingAverage2Raw(l,1) = mean(movingAverageRaw(m:(m-1)+500,1));
    movingAverage2Raw(l,2) = mean(movingAverageRaw(m:(m-1)+500,2));
    avg2_Mask(l,1) = mean(avg1_Mask(m:(m-1)+500,1));
    avg2_Machine(l,1) = mean(avg1_Machine(m:(m-1)+500,1));
    pressureDifference(l,1) = avg2_Machine(l,1) - avg2_Mask(l,1);
    m = m+1;
end
```

```
    if j == 1
        dataFor2s  = [avg2_Mask(l,1) avg2_Machine(l,1) pressureDifference(l,1) maskLeak(l,1)
machineOff(l,1)];
        assignin('base','dataFor2s',dataFor2s);
        time = timer;
        assignin('base','time',time);
        set(time,'executionMode','fixedRate');
        set(time,'Period',2);
        set(time,'TimerFcn','get2s(dataFor2s)');
        start(time);
    end


    if j >= 1000
        slopePS1(l,1) = avg2_Mask(l,1) - avg2_Mask(l-100,1);
        slopePS2(l,1) = avg2_Machine(l,1) - avg2_Machine(l-100,1);
        if(avg2_Machine(l,1)>=1)
            if(pressureDifference(l,1) > 0.3 && pressureDifference(l,1) < 1.3)
                maskLeak(l,1) = 1;
            elseif(pressureDifference(l,1) >= 1.3 && pressureDifference(l,1) < 2.5)
                maskLeak(l,1) = 2;
            elseif(pressureDifference(l,1) >= 2.5 && pressureDifference(l,1) < 5)
                maskLeak(l,1) = 3;
            elseif(pressureDifference(l,1) >= 5)
                maskLeak(l,1) = 4;
            end
        end
        if(avg2_Machine(l,1) < 1 && avg2_Mask(l,1) < 1)
            machineOff(l,1) = 1;
        end
    end

     if j>1
        dataFor2s = [avg2_Mask(l,1) avg2_Machine(l,1) pressureDifference(l,1) maskLeak(l,1)
machineOff(l,1)];
        assignin('base','dataFor2s',dataFor2s);
    end

    l = l+1;
    j = j+1;
    i = i+1;

    pause(0.001);

    assignin('base','rawMask',rawMask);
    assignin('base','rawMachine',rawMachine);
    assignin('base','maskPressure',maskPressure);
    assignin('base','machinePressure',machinePressure);
    assignin('base','movingAverage2Raw',movingAverage2Raw);
    assignin('base','maskLeak',maskLeak);
    assignin('base','slopePS1',slopePS1);
    assignin('base','slopePS2',slopePS2);
    assignin('base','machineOff',machineOff);
    assignin('base','avg1_Mask',avg1_Mask);
    assignin('base','avg1_Machine',avg1_Machine);
```

```
    assignin('base','avg2_Mask',avg2_Mask);
    assignin('base','avg2_Machine',avg2_Machine);
    assignin('base','pressureDifference',pressureDifference);
    assignin('base','dataFor2s',dataFor2s);
    assignin('base','l',l);
    assignin('base','j',j);
end
if b == 0
    pause(0.5);
    time = evalin('base','time');
    iter = evalin('base','iter');
    output = evalin('base','output');
    differentialTime = evalin('base','differentialTime');
    stop(time);

    leakIndex = [1 1];
    timeOfLeak = [0 0];
    leakIndex1 = [1 1];
    timeOfLeak1 = [0 0];
    n=2;
    percentPressure = 0;
    assignin('base','n',n);
    outputSize = size(output);
    leakTimePercent = 0;
    assignin('base','leakTimePercent',leakTimePercent);
    leakTimePercentI = 0;
    assignin('base','leakTimePercentI',leakTimePercentI);

    if outputSize(1,1) >= 15
        for n = 2:outputSize(1,1)
            if(output(n,4) == 0 && output(n-1,4) == 1)
                output(n-10:n-1,4) = 0;
                assignin('base','output',output);
            end
        end


         for n = 1:outputSize(1,1)-1
            if(output(n,4) == 0 && output(n+1,4) ~=0)
                leakIndex(co,1) = n;
                timeOfLeak(co,1) = differentialTime(n,1);
                co = co+1;
            end
            if(output(n,4) ~= 0 && output(n+1,4) ==0)
                leakIndex(co1,2) = n+1;
                timeOfLeak(co1,2) = differentialTime(n+1,1);
                co1 = co1+1;
            end
         end

        numberOfLeaks = size(leakIndex);

        for n = 1:numberOfLeaks(1,1)
            if(leakIndex(n,2) == 0 || (leakIndex(n,1) ~= 1 && leakIndex(n,2) == 1))
                leakIndex(n,2) = iter;
                timeOfLeak(n,2) = differentialTime(iter);
```

```
    end
end
assignin('base','leakIndex',leakIndex);
assignin('base','timeOfLeak',timeOfLeak);


for n = 1 : numberOfLeaks(1,1)
    percentPressure(n,1) = max(output(leakIndex(n,1):leakIndex(n,2),4))*25;
end
assignin('base','percentPressure',percentPressure);
assignin('base','numberOfLeaks',numberOfLeaks);

sumOfLeaks = 0;
for n = 15:outputSize(1,1)
    if(output(n,4) ~= 0)
        sumOfLeaks = sumOfLeaks + 1;
        assignin('base','sumOfLeaks',sumOfLeaks);
    end
end
leakTimePercent = (sumOfLeaks/(iter-15+1))*100;
assignin('base','leakTimePercent',leakTimePercent);



% -- Instrument ------ %

for n = 1:outputSize(1,1)-1
    if(output(n,5) == 0 && output(n+1,5) ~=0)
        leakIndex1(col,1) = n+1;
        timeOfLeak1(col,1) = differentialTime(n+1,1);
        col = col+1;
    end
    if(output(n,5) ~= 0 && output(n+1,5) ==0)
        leakIndex1(col1,2) = n+1;
        timeOfLeak1(col1,2) = differentialTime(n+1,1);
        col1 = col1+1;
    end
 end

numberOfLeaksI = size(leakIndex1);
assignin('base','numberOfLeaksI',numberOfLeaksI);
for n = 1:numberOfLeaksI(1,1)
    if(leakIndex1(n,2) == 0 || (leakIndex1(n,1) ~= 1 && leakIndex1(n,2) == 1))
        leakIndex1(n,2) = iter;
        timeOfLeak1(n,2) = differentialTime(iter);
    end
end
if leakIndex1(1,1) > 12
    for n = 1:numberOfLeaksI(1,1)
        output(leakIndex1(n,1)-12:leakIndex1(n,1),5) = 1;
    end
    leakIndex1(:,1) = leakIndex1(:,1) - 12;
    timeOfLeak1(:,1) = timeOfLeak1(:,1) - 24;
end
```

```
        assignin('base','leakIndex1',leakIndex1);
        assignin('base','timeOfLeak1',timeOfLeak1);



        sumOfLeaksI = 0;
        for n = 30:outputSize(1,1)
            if(output(n,5) == 1)
                sumOfLeaksI = sumOfLeaksI + 1;
                assignin('base','sumOfLeaksI',sumOfLeaksI);
            end
        end
        leakTimePercentI = (sumOfLeaksI/(iter-30+1))*100;
        assignin('base','leakTimePercentI',leakTimePercentI);


        timeOfLeak;
        percentPressure;
        leakTimePercent;
        timeOfLeak1;
        leakTimePercentI;
    end

%% EXPORT TO EXCEL
    filepath = 'C:\Users\aishu\Dropbox\UTARI\DATA\dataCollect\leakDetectorData_';
    fileext = '.xlsx';
    filename = strcat(filepath,docstr,fileext);
    assignin('base','filename',filename)
    dataHeading = {'Date/Time','dt','PS1','PS2','Pressure Difference','Mask Leak','Machine off'};
    dataBody = [differentialTime,output(:,1),output(:,2),output(:,3),output(:,4),output(:,5)];
    dataCell = num2cell(dataBody);
    assignin('base','dataCell',dataCell);
    date = evalin('base','date');
    dataWithDate = [date,dataCell];
    finalDataToExcel = [dataHeading;dataWithDate];
    xlswrite(filename,finalDataToExcel);
    disp('Stopped');
    reportPath = 'C:\Users\aishu\Dropbox\UTARI\DATA\dataCollect\Leak_Detector_Report_';
    reportPathFull = strcat(reportPath,docstr);

publish('C:\Users\aishu\Dropbox\UTARI\programs\new\Leak_Detector_Report.m','format','pdf','showCode'
,false,'outputDir',reportPathFull);
    clear all;
end

%% NOTIFICATION METHOD
% --- 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 = cellstr(get(hObject,'String')) returns popupmenu1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu1
contents = get(hObject,'Value');
```

```
switch contents
   case 1
      option = 0;
      assignin('base','option',option);
   case 2
      option = 1;
      assignin('base','option',option);
   case 3
      option = 2;
      assignin('base','option',option);
   otherwise
      option = 0;
      assignin('base','option',option);
end


% --- 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


% Timer function to record the data every 2s

function [ z ] = get2s(x)
% RECORD DATA EVERY 2s
%% ---- VARIABLE DECLARATION ------ %%
persistent iter;
persistent saveData;
persistent date;
persistent differentialTime;
persistent count;
persistent var;
persistent percentLeak;
persistent percentLeakstr;
persistent s1;
persistent s2;
persistent ai;
persistent leakTime;
persistent message;
persistent y;
persistent Fs;
persistent aud;
persistent start;
persistent stop;
persistent maxLeak;
persistent bi;
persistent count1;
persistent var1;
```

```
persistent sI;
persistent message1;
persistent leakTime1;
if isempty(iter)
   iter = 1;
   date = cell(1);
   count = 0;
   var = 0;
   var1 = 0;
   percentLeak = 0;
   percentLeakstr = '0';
   ai = 1;
   maxLeak = 0;
   bi = 1;
   count1 = 0;
end
%% ----- RECORD DATA AND LEAK NOTIFICATION--------- %%
saveData(iter,1:5) = x;
date(iter,1) = {datestr(now)};
differentialTime(iter,1) = (iter-1)*2;

if((saveData(iter,4) == 1 || saveData(iter,4) == 2 || saveData(iter,4) == 3 || saveData(iter,4) == 4 )&& var
== 0)
    if ai == 1
        leakTime = char(date(iter,1));
        assignin('base','leakTime',leakTime);
        ai = 0;
    end
    count = count + 1;
    if count > 5
        maxLeak = max(maxLeak,saveData(iter,4));
    end
end

if(saveData(iter,5)==1 && var1 == 0)
    if bi == 1
        leakTime1 = char(date(iter,1));
        assignin('base','leakTime1',leakTime1);
        bi = 0;
    end
    count1 = count1 + 1;
end

option = evalin('base','option');
if count == 10
    if(option == 1)
        percentLeak = (100/4)*maxLeak;
        percentLeakstr = num2str(percentLeak);
        s1 = 'Location: Mask Side. Maximum Leak Percentage is  ';
        s2 = ' Start time of the Leak is ';
        message = strcat(s1,percentLeakstr,s2,leakTime);
        assignin('base','message',message);
        send_text_message('6825538604','T-Mobile','Leak',message);
    end
    if(option == 2)
        [y,Fs] = audioread('C:\Users\aishu\Dropbox\UTARI\programs\new\Thalli Pogathey.mp3');
```

```
        aud = audioplayer(y,Fs);
        start = aud.SampleRate * 0.1;
        stop = aud.sampleRate * 14;
        play(aud,[start stop]);
        disp('leak');
    end
     count = 0;
    var = 1;
end
if count1 == 10
    if(option == 1)
        sI = 'Location : Instrument Side. Start time of the leak is ';
        message1 = strcat(sI,leakTime1);
        assignin('base','message1',message1);
        send_text_message('6825538604','T-Mobile','Leak',message1);
    end
    if(option == 2)
        [y,Fs] = audioread('C:\Users\aishu\Dropbox\UTARI\programs\new\Thalli Pogathey.mp3');
        aud = audioplayer(y,Fs);
        start = aud.SampleRate * 0.1;
        stop = aud.sampleRate * 14;
        play(aud,[start stop]);
        disp('off');
    end
    count1 = 0;
    var1 = 1;
end
if(saveData(iter,4) == 0)
    var = 0;
    count = 0;
    ai = 1;
    leakTime = '0';
end

if(saveData(iter,5) == 0)
    var1 = 0;
    count1 = 0;
    bi = 1;
    leakTime1 = '0';
end
assignin('base','output',saveData);
assignin('base','iter',iter);
assignin('base','date',date);
assignin('base','differentialTime',differentialTime);
assignin('base','count',count);
assignin('base','count1',count1);
output = saveData(iter,:);
iter = iter+1;
end

%    Function to send message notification

function send_text_message(number,carrier,subject,message)
% SEND_TEXT_MESSAGE send text message to cell phone or other mobile device.
%    SEND_TEXT_MESSAGE(NUMBER,CARRIER,SUBJECT,MESSAGE) sends a text message
%    to mobile devices in USA. NUMBER is your 10-digit cell phone number.
```

```
%    CARRIER is your cell phone service provider, which can be one of the
%    following: 'Alltel', 'AT&T', 'Boost', 'Cingular', 'Cingular2',
%    'Nextel', 'Sprint', 'T-Mobile', 'Verizon', or 'Virgin'. SUBJECT is the
%    subject of the message, and MESSAGE is the content of the message to
%    send.
%
%    Example:
%      send_text_message('234-567-8910','Cingular', ...
%         'Calculation Done','Don't forget to retrieve your result file')
%      send_text_message('234-567-8910','Cingular', ...
%         'This is a text message without subject')
%
%   See also SENDMAIL.
%
% You must modify the first two lines of the code (code inside the double
% lines) before using.

% Ke Feng, Sept. 2007
% Please send comments to: jnfengke@gmail.com
% $Revision: 1.0.0.0 $  $Date: 2007/09/28 16:23:26 $

% ========================================================================
% YOU NEED TO TYPE IN YOUR OWN EMAIL AND PASSWORDS:
mail = 'leak.detector01@gmail.com';    %Your GMail email address
password = '********';                 %Your GMail password
% ========================================================================

if nargin == 3
   message = subject;
   subject = '';
end

% Format the phone number to 10 digit without dashes
number = strrep(number, '-', '');
if length(number) == 11 && number(1) == '1';
   number = number(2:11);
end

% Information found from
% http://www.sms411.net/2006/07/how-to-send-email-to-phone.html
switch strrep(strrep(lower(carrier),'-',''),'&','')
   case 'alltel';    emailto = strcat(number,'@message.alltel.com');
   case 'att';       emailto = strcat(number,'@mmode.com');
   case 'boost';     emailto = strcat(number,'@myboostmobile.com');
   case 'cingular';  emailto = strcat(number,'@cingularme.com');
   case 'cingular2'; emailto = strcat(number,'@mobile.mycingular.com');
   case 'nextel';    emailto = strcat(number,'@messaging.nextel.com');
   case 'sprint';    emailto = strcat(number,'@messaging.sprintpcs.com');
   case 'tmobile';   emailto = strcat(number,'@tmomail.net');
   case 'verizon';   emailto = strcat(number,'@vtext.com');
   case 'virgin';    emailto = strcat(number,'@vmobl.com');
end

%% Set up Gmail SMTP service.
% Note: following code found from
% http://www.mathworks.com/support/solutions/data/1-3PRRDV.html
```

```
% If you have your own SMTP server, replace it with yours.

% Then this code will set up the preferences properly:
setpref('Internet','E_mail',mail);
setpref('Internet','SMTP_Server','smtp.gmail.com');
setpref('Internet','SMTP_Username',mail);
setpref('Internet','SMTP_Password',password);

% The following four lines are necessary only if you are using GMail as
% your SMTP server. Delete these lines wif you are using your own SMTP
% server.
props = java.lang.System.getProperties;
props.setProperty('mail.smtp.auth','true');
props.setProperty('mail.smtp.socketFactory.class', 'javax.net.ssl.SSLSocketFactory');
props.setProperty('mail.smtp.socketFactory.port','465');

%% Send the email
sendmail(emailto,subject,message)

if strcmp(mail,'matlabsendtextmessage@gmail.com')
    disp('Please provide your own gmail for security reasons.')
    disp('You can do that by modifying the first two lines of the code')
    disp('after the bulky comments.')
end
```

Appendix B

Connector Design

Design 1:

Design 1 of the custom connector was designed to create a bridge that connects the PAP (Positive Airway Pressure) mask to the hose, and the hose to the PAP machine. The bridge houses a pressure and microphone sensor. The device consists of a center piece, hose connector, and mask connector. The hose connector is the end of the device that connects to the hose, and the mask connector is the end that connects to the mask or PAP machine. The bridge is 3D printed using an Objet500 Connex3 3D printer. The center piece and hose connector are 3D printed with a hard, plastic-like material called VeroWhite. The mask connector is made of a softer material, which is a combination of the materials VeroWhite and TangoBlack. This material has some elasticity, and is able to create a tight fit around the mask connection, without slipping.

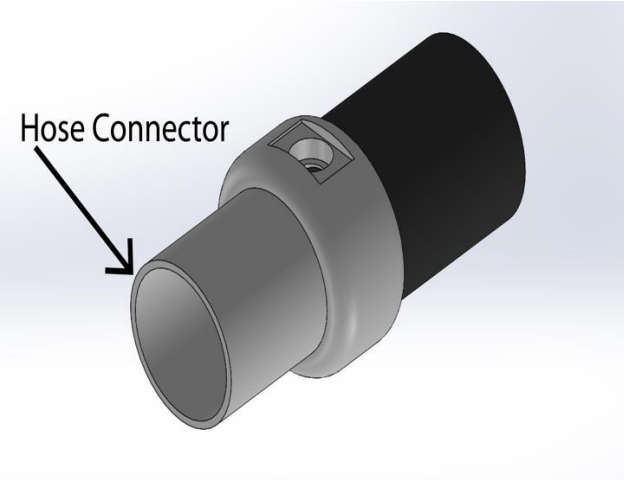


Figure 1. *View of the hose connector.*

The arrow in Figure 1 points towards the hose connector. This piece fits into the hose used on PAP machines. It has an outer diameter of 22.10 mm, and an inner diameter of 20.00 mm. This creates a tight fit between the custom connector and the PAP hose. The length of the hose connector is 27.00 mm.

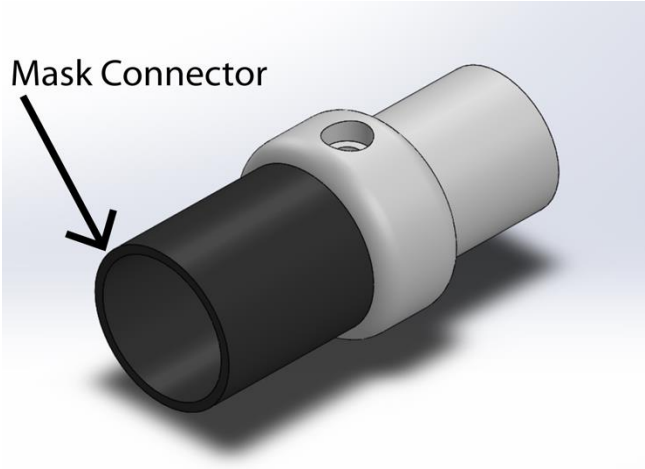


Figure 2. *View of the mask connector.*

Figure 2 depicts the mask connector, which connects either to the PAP machine or the mask. The inner diameter is 22.10 mm. This creates a tight fit that also allows no slipping when the mask is inserted or when the connector is inserted onto the PAP machine. This piece is 21.00 mm long, which allows for the mask or PAP machine to be inserted fully as if the hose was being attached instead.

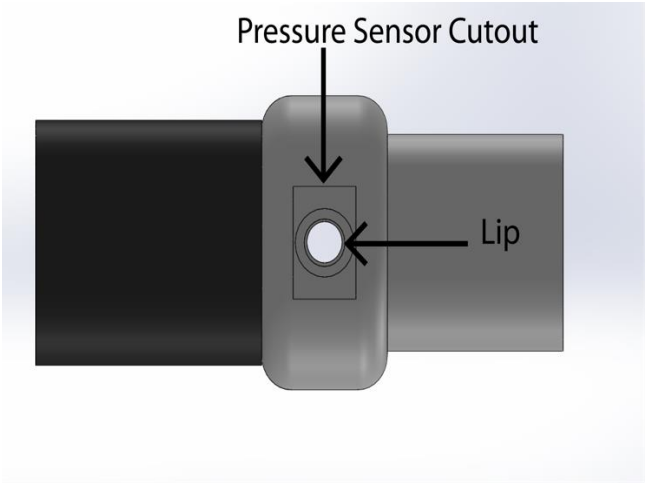


Figure 3. *View of pressure sensor cutout.*

The rectangular cutout for the pressure sensor, as shown in Figure 3, is 11.50 by 7.50 mm. The cutout is 1.00 mm deep to allow the sensor to sit flush with the body. The circular cutout for the

circular part of the pressure sensor has a diameter of 7.00 mm. There is a lip with a diameter of 4.86 mm added inside of this feature to add support to the pressure sensor and to keep it from entering the lumen.

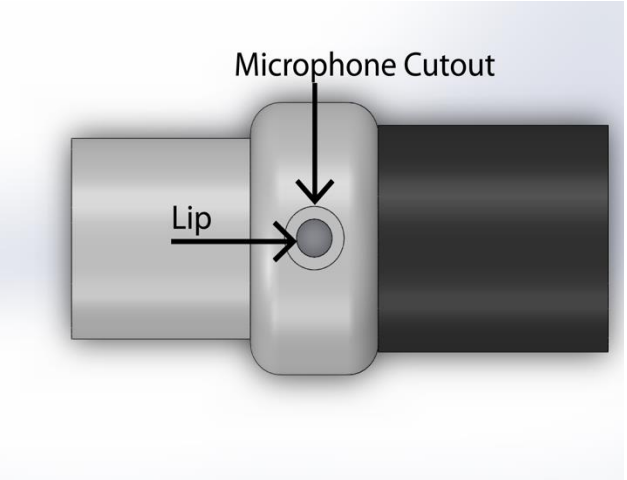


Figure 4. *View of the microphone cutout.*

The cutout for the microphone sensor has a 7.00 mm diameter, with a depth of 3.50 mm. There is a lip with a diameter of 4.20 mm to secure the microphone in place so it cannot enter the lumen.

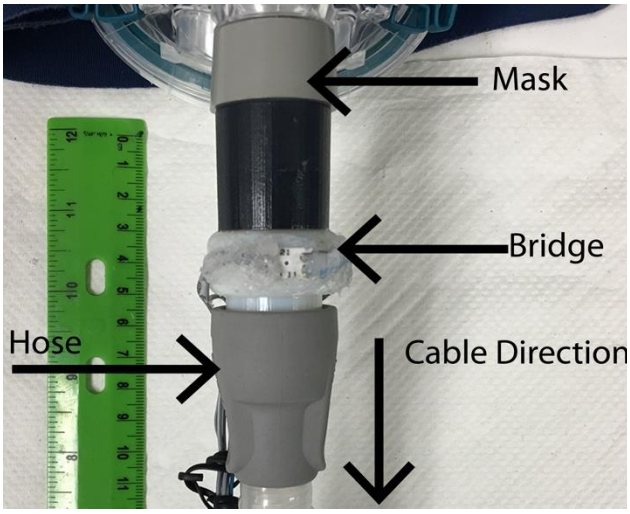


Figure 5. *Bridge attached to mask.*

Figure 5 shows the connector attached to the mask and hose. The mask is inserted into the connector, and the connector is inserted into the hose. The cable runs in the direction of the hose towards the PAP machine.

Design 2:

The redesign of the custom connector addressed some key issues from the first design and adds functionality. The redesign allows for a ribbon cable to be plugged into the connector to make tasks such as cleaning and storing less cumbersome. This new version also features new pressure and microphone sensors that are significantly larger than the previous version's. The redesign places the new microphone and pressure sensors in more convenient locations that put them out of the way for easier handling by users. The redesign of the connector varies based on whether it will be attached to the mask or the PAP machine. Regardless of which connector is being used, the 3 basic sections that are present in design 1 are also present here: a mask connector, a center piece, and a hose connector. The center piece of each connector has detachable parts that makes getting sensors in and out a much more time-friendly task that will minimize damage to the connector and sensors.

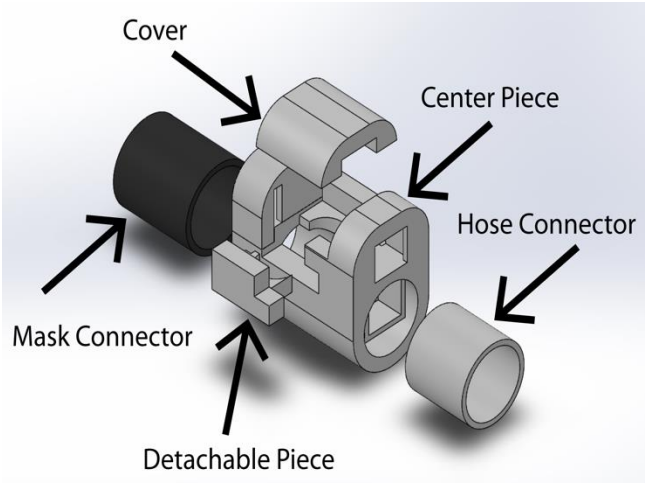


Figure 6. *Exploded view of connector designed to connect to PAP machines.*

As figure 6 shows, the custom connector is divided into 3 sections; mask connector, center

piece, and hose connector. The hose connector is the same as in design 1 while the mask connector has an inner diameter of 21.50 mm. The center piece consists of various pieces that fit together to keep the microphone, pressure sensors and ribbon cable connector in place. The total length of the connector is 93.00 mm, with a height of 28.46 mm at its tallest point, and a 20.00 mm diameter lumen. Inside the lumen there is a cage that creates an obstruction of airflow that helps the pressure sensor more accurately depict pressure changes within the system.

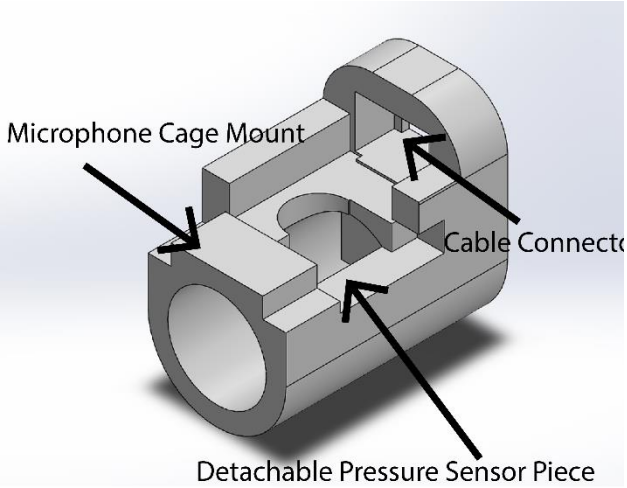


Figure 7. *Center piece of connecter designed to attach to a PAP machine.*

Figure 7 depicts the center piece. The center piece consists of multiple pieces that come together to keep the microphone and pressure sensors in place. The microphone cage mount is where the microphone cage with a microphone is mounted. Making it removable allows for the microphone to be easily mounted, and replaced if needed. The microphone diaphragm faces the PAP machine to audibly detect fine leaks. The detachable pressure sensor piece allows for the pressure sensor to be easily mounted and removed without damaging the sensor or the connector. The ribbon cable connector gets placed in the area labeled cable connector. The cutout for the connector is 15.00 mm by 10.50 mm by 12.25 mm, allowing for a snug fit.

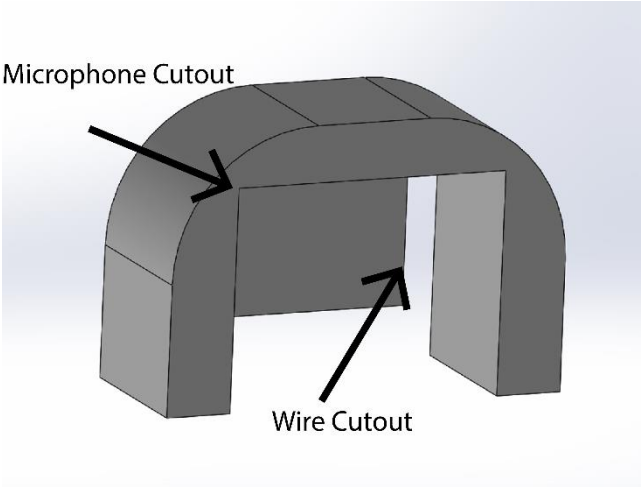


Figure 8. *Microphone cage.*

The microphone cage houses the outside microphone. The microphone cutout is 11.00 mm by 19.00 mm with a depth of 9.00 mm. The wire cutout is there to guide wires from the microphone to the cable connector to be soldered. The microphone cage mounts onto the microphone cage mount in Figure 7.

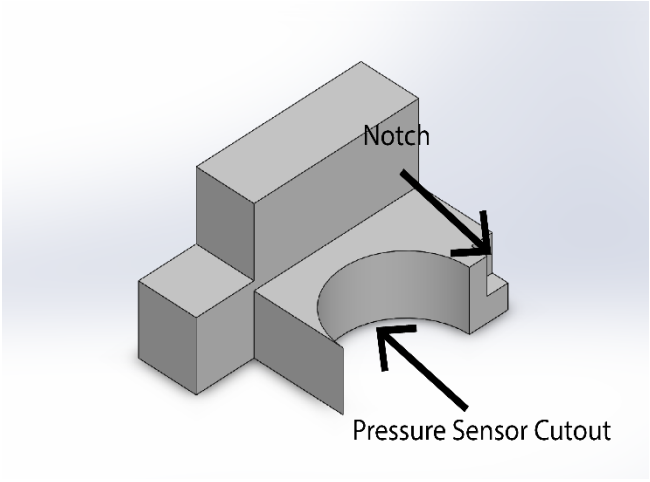


Figure 9. *Detachable Pressure Sensor Piece.*

Figure 9 depicts the detachable pressure sensor piece. The detachable pressure sensor piece attaches the pressure sensor to the center piece. The pressure sensor cutout, when connected to the center piece, has a diameter of 12.00 mm to

fit the pressure sensor. The piece fits snugly into the center piece, but glue is used to create a perfect seal to make sure no leak is present. The notch allows for easy detachment from the center piece by prying. This prevents the center piece or pressure sensor from being damaged upon removal.

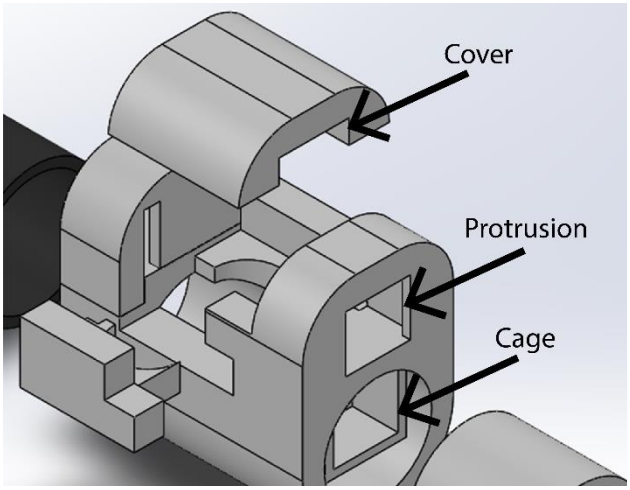


Figure 10. *Center Piece Zoomed in.*

Figure 10 shows a closer view of the center piece. The center piece has a length of 45.00 mm. The center piece has a cover piece that helps keep the cable connector in place and provides protection for the pressure sensor. The cover has a length of 24.20 mm with a gap that is 4.43 mm in height and 13.00 mm wide. When placed on the center piece, it will provide support for the ribbon cable connector when a cable is plugged in. The protrusion sticks out 1.50 mm from both sides, leaving an opening of 12.00 mm, and prevents the ribbon cable connector from sliding out forward. The cage is 15.00 mm by 11.22 mm with a width of 11.50 mm. The cage itself has a hollowed out section in the center to save material when 3D printing. This creates an environment that allows the pressure sensor to accurately detect small changes of pressure within the system. Without the cage present in the lumen, the pressure sensor fails to detect minute pressure changes that are critical for detecting fine leaks on a patient's mask.

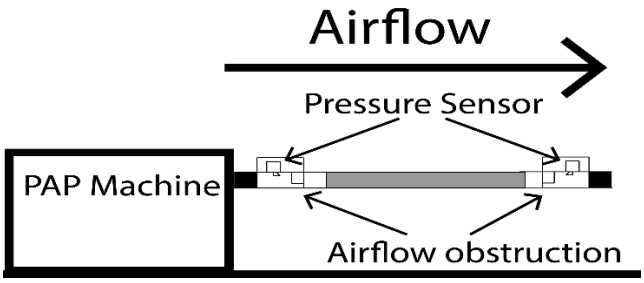


Figure 11. *Diagram depicting airflow and placement of airflow obstructions.*

Figure 11 depicts how air travels from the PAP machine through the system. The pressure sensor on the left belongs to the bridge connected to the PAP machine, and the pressure sensor on the right depicts the pressure sensor on the bridge connected to the mask (not illustrated).

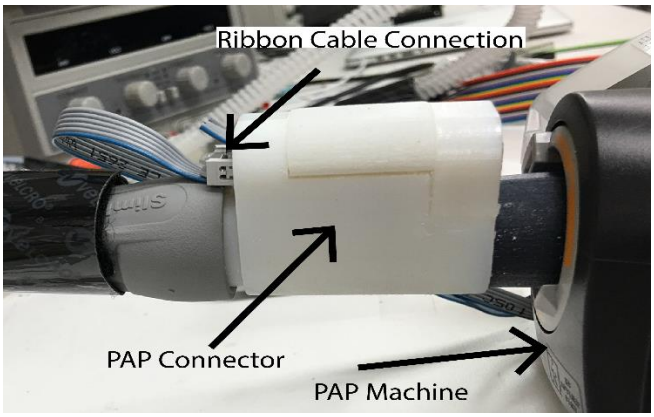


Figure 12. *PAP Connector Attached to PAP machine*

Figure 12 shows the PAP connector attached to the PAP machine. The ribbon cable is attached to the connector via the ribbon cable connector. The cable can be easily attached and removed as needed to handle to the device. On the opposite end, the microphone (not shown in the picture) is facing the PAP machine.

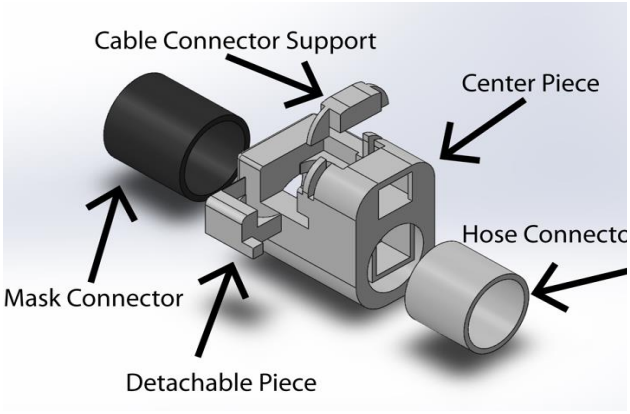


Figure 13. *Exploded view of connector designed to attach to a mask.*

The connector shown in Figure 13 is designed to be attached to a mask. The connector has the same basic design as the one depicted in Figure 6. It has a mask connector, hose connector, and a center piece with multiple parts. The hose and mask connectors are the same as the ones used in Figure 6. The center piece consists of different parts that fit together to keep the pressure sensor and ribbon cable connector in place. Unlike the connector in Figure 6, this connector does not have a microphone. The total length of the connector is 91.00 mm, with a height of 40.50 mm at its tallest point. The lumen has a diameter of 20.00 mm. Just like the connector in Figure 6, this connector design also has a cage in the lumen to obstruct airflow.

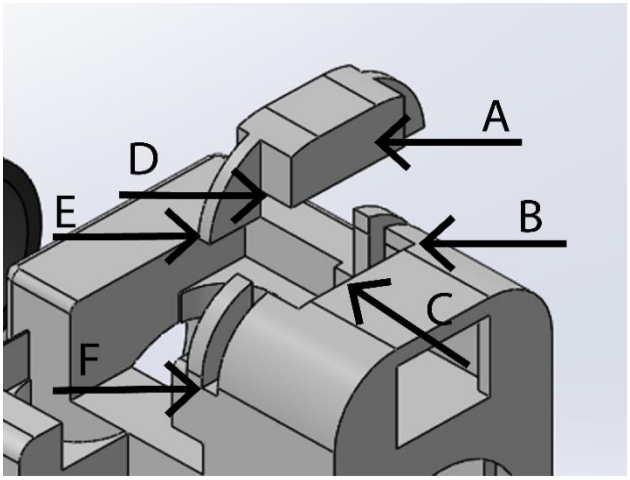


Figure 14. *Support for cable connector.*

Figure 14 depicts how the cable connector support fits into the center piece. This piece supports the ribbon cable connector when a cable is being plugged in. Label A has a width of 14.40 mm, that can fit into where Label C points to. Label D has a width of 3.50 mm, that fits next to label B. Label E has a width of 1.50 mm and fits into label F. The gap present under label A, 14.40 mm wide and a height of 3.00 mm, allows for soldered connections from the cable connector to the pressure sensor.

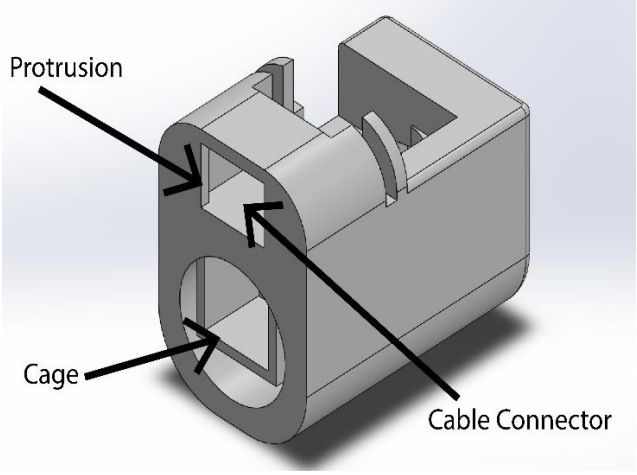


Figure 15. *Mask Connector Center Piece.*

The cage shown in figure 15 is 15.00 mm by 11.22 mm with a width of 11.50 mm. The cage itself has a hollowed out section in the center to save material when 3D printing. The ribbon cable connector rests behind a 12.00 mm wide opening. The opening is large enough to allow a ribbon cable to be plugged in, but does not allow the ribbon cable connector to slip out of the center piece when the ribbon cable is disconnected due to 1.50 mm protrusions on both sides.

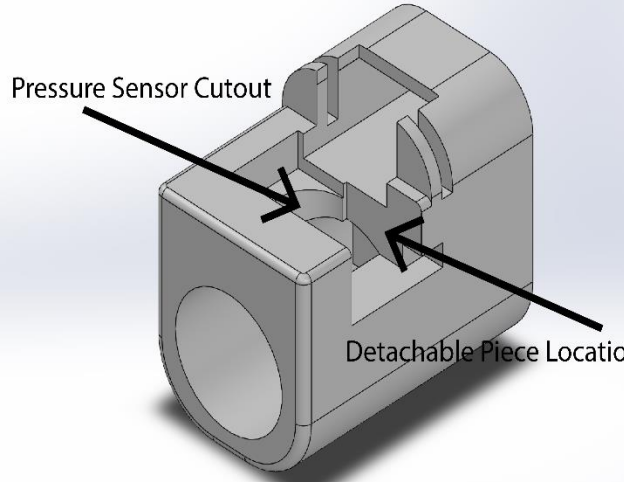


Figure 16. *Pressure Sensor Cutout.*

Figure 16 shows the cutouts for the pressure sensor. The pressure sensor cutout, when combined with the detachable piece, has a rectangular cutout measuring 12.50 mm by 17.00 mm. The depth of this cutout is 5.00 mm, which gives the pressure sensor protection when the device is being handled. Inside of the rectangular cutout is another cutout that allows the pressure sensor to reach the lumen. This cutout has a diameter of 12.00 mm with a depth of 5.00 mm. The detachable piece allows for the pressure sensor to be mounted and unmounted from the connector with ease.

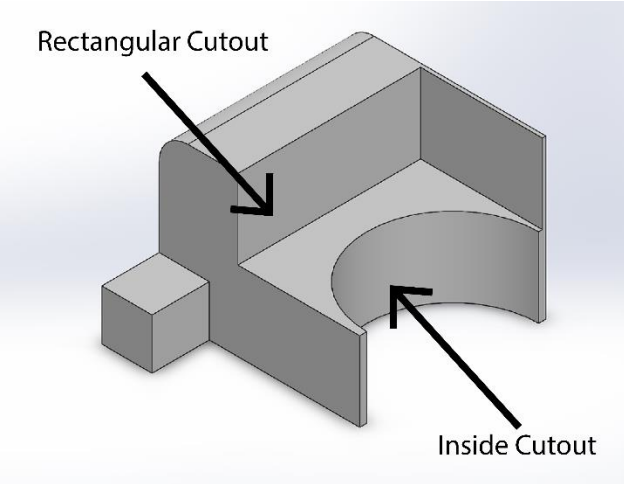


Figure 17. *Detachable piece*

The detachable piece fits into the center piece. The piece completes a 12.50 mm by 17.00 mm rectangular cutout with a 12.00 mm diameter cutout in the center. The detachable piece fits snugly into the center piece with the pressure sensor. To ensure a complete seal, glue is used to hold the parts together.

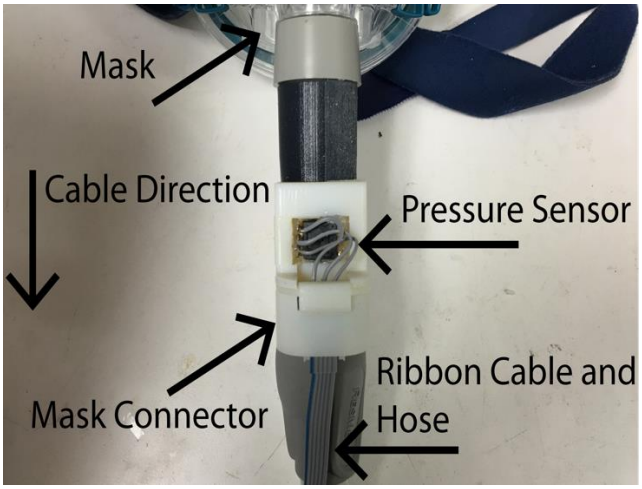


Figure 18. *Mask Connector attached to mask and hose.*

Figure 18 shows the mask connector attached to the mask and the hose. The mask fits into the bridge, and the hose connector fits into the hose. In this configuration, the ribbon cable will follow the hose away from the mask. The airflow is the opposite of the cable direction.

Appendix C

Generated Report

LEAK DETECTOR SYSTEM REPORT

Table of Contents

Information

Resmed Airsense 10

20-Jul-2016 14:18:52

Overall Performance

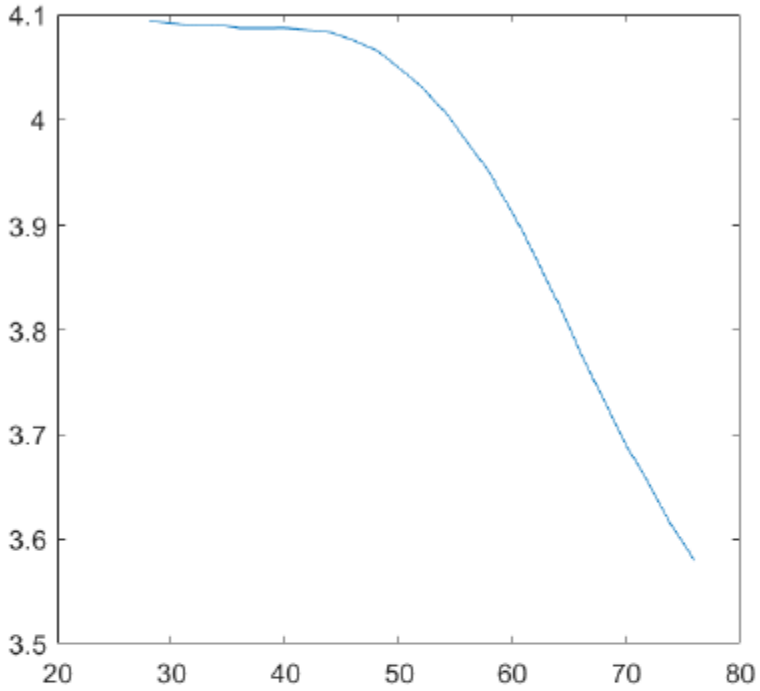
The system is checked for leaks and the percentage of leaks during the run is given below

Leaks were present less than 50 percent of the entire run time

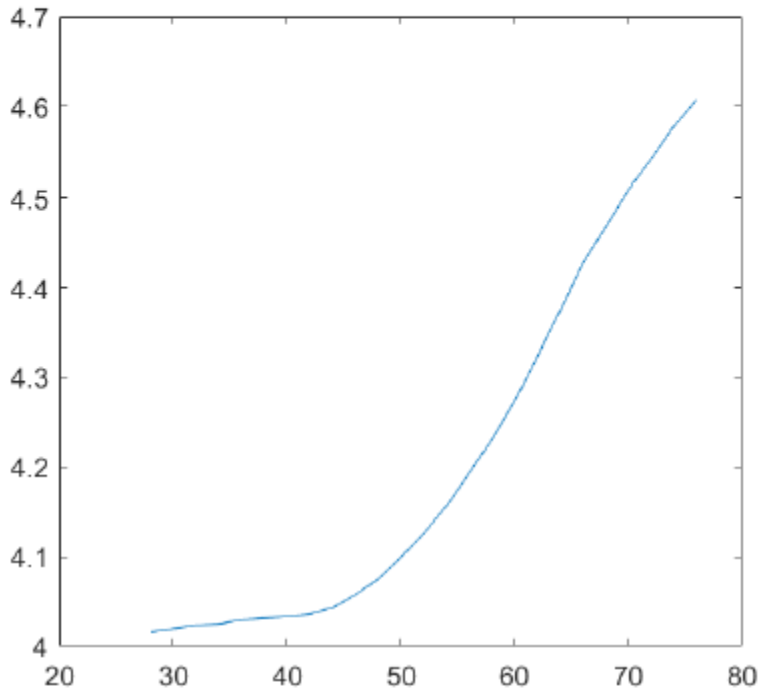
Pressure Changes

The pressure varies according to the pressure being set by the clinician The pressure ranges between 4 - 20 cm of water

Mask Side Pressure



Machine Side Pressure:



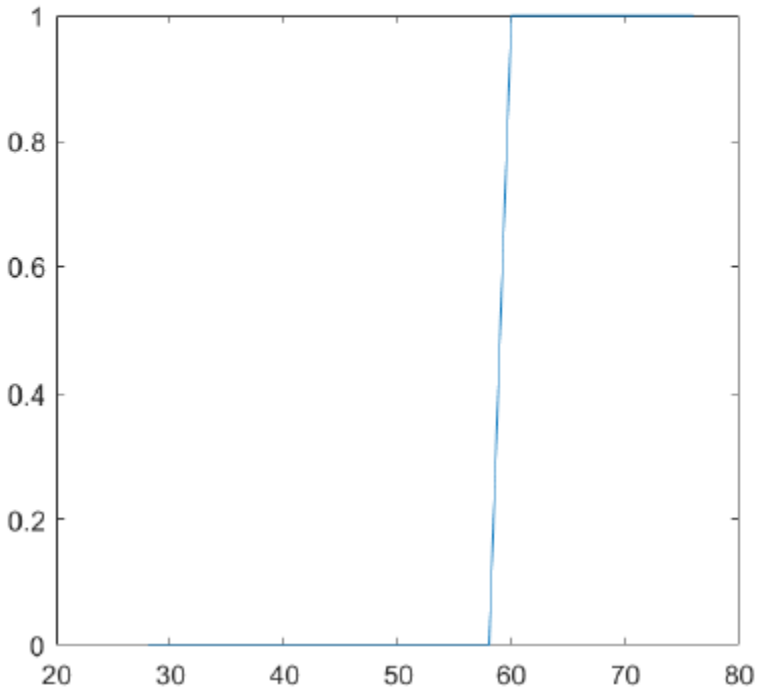
Leak Information

Over the entire run of the system, PAP machine is checked for leaks and classified as mask leak and machine leak.

Mask Leak

Mask leak denotes the leaks occuring on the mask side of the PAP machine

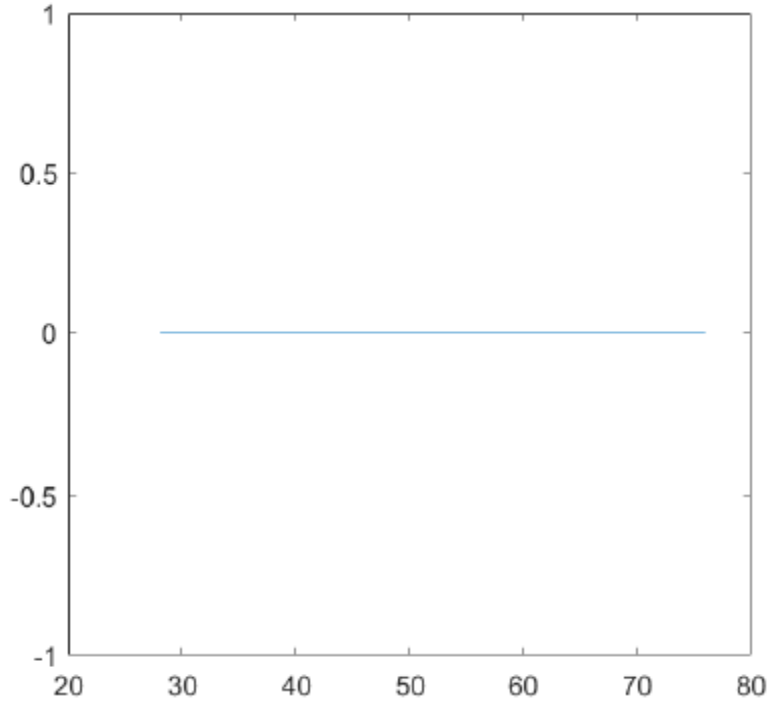
```
The leak occured between time20-Jul-2016 14:17:36  and20-Jul-2016
 14:17:54 .The Intensity of Leak during this time is25  percent


The total percentage of mask leak during the entire run is
    36
```



Machine Leak

Mask leak denotes the leaks occuring on the mask side of the PAP machine

```
The total percentage of machine leak during the entire run is
     0
```



Published with MATLAB® R2015b

References

1. Parkes, J.D. Sleep and its disorders. W.B. Saunders Company. 1985; p.335 14. Major problems in neurology. 0-7216-1858-8.
2. Phillipson EA. Sleep apnea: a major health problem (editorial). N Engl J Med 1993;328: 1271-3.
3. Massimo R. Mannarino, Francesco Di Filippo, Matteo Pirro Obstructive Sleep Apnea Syndrome, European Journal of Internal Medicine 2012; 586-593.
4. Proefschrift Obstructive Sleep Apnea Syndrome Pathogenic Aspects and Treatment Thesis Dissertation Erasmus University Rotterdam, 2000
5. Lilian Kepkemboi Kigen, Long term use of Continuous Positive Airway Pressure care in sleep apnea: Literature review on common side effects and possible nursing interventions Thesis Dissertation, Central Ostrobothnia University of Applied Science, November 2011.
6. David P. Kuehn, Continuous Positive Airway Pressure in the treatment of Hypernasality, American Journal of Speech-Language Pathology, November 1997, Vol. 6, 5-8. doi:10.1044/1058-0360.0604.05
7. Giles TL1, Lasserson TJ, Smith BJ, White J, Wright J, Cates CJ, Continuous positive airways pressure for obstructive sleep apnoea in adults, Cochrane Database Syst Rev. 2006 Jan 25;(1):CD001106.
8. Hedner, J., Grote, L. & Zou, D. Pharmacological treatment of sleep apnea: Current situation and future strategies. 2007. Sleep Laboratory, Department of Pulmonary Medicine and Allergology, Sahlgrenska Hospital, 413 45 Gothenburg, Sweden.
9. Engleman, H. M., Kingshott, R.N., Wraith, P. K., Mackay, T.W, Deary, I. J & Douglas, N. J. 1999. Randomized Placebo-controlled Crossover Trial of Continuous Positive Airway Pressure for Mild Sleep Apnea/Hypopnea Syndrome. 1999. Departments of Respiratory Medicine and Psychology, University of Edinburgh, United Kingdom.
10. National Institutes of Health. National Heart, Lung and Blood Institute. CPAP. 2010. Available: http://www.nhlbi.nih.gov/health/health-topics/topics/cpap/risks.html. Accessed 2 September 2011.
11. National Sleep Foundation. 2006. Sleep and CPAP Adherence. Available: http://www.sleepfoundation.org/article/ask-the-expert/sleep-and-cpap-adherence. Accessed 15 July 2010.
12. Alicia M. Gibb, New media art, design, and the Arduino microcontroller: a malleable tool Thesis Dissertation School of Art and Design Pratt Institute, February 2000.

13. G.Schmalisch, H.Fischer, C.C.Roehr, H.Proquitte, Comparison of different techniques to measure air leaks during CPAP treatment in neonates, medengphy, 2008.05.002
14. Xuanju Shang, Development of a hydraulic component leakage detecting system using pressure decay signal, University of Northern Iowa, 2015
15. A. Tzavaras ; Dept. of Med. Instrum. Technol., Technol. Educ. Inst. of Athens, Aigaleo, Greece ; B. Spyropoulos ; P. R. Weller, Fuzzy reasoning Clinical Decision Support for manual titration of Positive Airway Pressure Support and Oxygen supply in patients with Obstructive Sleep Apnea, IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI), June 2014

Biographical Information

Aishwarya Gopalakrishnan was born on 27th of June 1992, in Chennai India. She completed her Bachelors of Engineering in Electronics and Instrumentation Engineering at SRM Easwari Engineering college, Chennai India in May 2013. She started her career at Infosys as Systems Engineer. She underwent her training at Infosys Mysore campus, India. She was working in the Software field and learnt the programming languages. She then left her job to pursue her Master's degree. She started her graduate studies in Fall 2014 at the University of Texas at Arlington. In Spring 2015, she was appointed as a Graduate Teaching Assistant for the course Mathematical Foundations of Electrical Engineering where she assisted Dr. Kambiz Alavi. As she was very much interested in Research, she applied for positions related to her field of interest, Embedded systems and Algorithm Development at The University of Texas at Arlington Research Institute and she started her internship at UTARI for the Leak Detector System for Sleep Apnea project and then she took this project as her thesis project and designed algorithm for the project. She plans to pursue her interests by taking up challenging projects and developing algorithms for those.