A Healthcare IoT Prototype for Responsive Oxygen Therapy Treatment of COPD Patients

By

Azer Khan

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

in

The Department of Computer Science

State University of New York

New Paltz, New York 12561

December 2020
A HEALTHCARE IOT PROTOTYPE FOR RESPONSIVE OXYGEN THERAPY TREATMENT OF COPD PATIENTS

Azer Khan
State University of New York at New Paltz

We, the thesis committee for the above candidate for the Master of Science degree, hereby recommend acceptance of this thesis.

Chirakkal Easwaran, Thesis Advisor
Department of Computer Science, SUNY New Paltz

Rajeev Narayanan, Thesis Committee Member
Director of Ecosystem Platform Strategy, Eisai Incorporated

Damodaran Radhakrishnan, Thesis Committee Member
Department of Computer Science, SUNY New Paltz

Submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science at the State University of New York at New Paltz
Abstract

Our final design offers oxygen therapy patients an IoT enabled, adaptable and small form-factor device offering potential for automatic detection and agile response to oxygen saturation readings. Our key components are a photodiode sensor, algorithm processor and micro-controller providing the foundation for future development for FDA approval, machine-learning and analytics, and feedback-loop oxygen tank controller tracks. Our device is cutting-edge in its communications, power consumption and efficiency.

We have gained an understanding of the effort required to design an IoT enabled solution in the healthcare space. Integrating hardware and software designs are an exercise to understand the inner workings of many systems used today. It is estimated that there will be about 50 billion IoT-enabled devices in the world by 2030 [22]. Technologists who understand the underlying systems will be able to make well-informed decisions about the future of the connected world.
Table of Contents

Abstract 3
List of Tables 7
List of Figures 8
List of Equations 9

1. Introduction 1
   1.1 Goals of this Study 2
   1.2 Background 2
   1.3 The Circulatory System and Oxygen Transport 2
   1.4 Tissue Oxygenation 2
   1.5 The Respiratory System and Oxygen Transport 4
   1.6 Binding of Oxygen to Hemoglobin: Oxygen Saturation Curve 4
   1.7 Gas Laws: Dalton’s Law and Henry’s Law 5
      1.7.1 Exchange of Oxygen and Carbon Dioxide 5

2. Pulse Oximetry 6
   2.2 Principles of SpO₂ Measurement 6
   2.3 Beer-Lambert Law in Oximetry 9
   2.4 Theory of Pulse Oximetry 11
   2.5 Why is Calibration Required? 14

3. Related Work 16
   3.1 Pulse Oximeter from Texas Instruments 16
   3.2 Pulse Oximeter from Arrow (and Microchip Technologies) 17
   3.3 Pulse Oximeter from Analog 18

4. Product Management (Design) 19
   4.1 Wireless Communication 20
   4.2 ZigBee (IEEE 802.15.4) 20
   4.3 Bluetooth (IEEE 802.15.1) 21
   4.4 Sparkfun Pulse Oximeter and Heart Rate Sensor 21
      4.4.1 Power and I²C Pins: 22
      4.4.2 Additional Pins: 23
      4.4.3 MAX30101 - Pulse Oximeter and Heart Rate Monitor 23
      4.4.4 MAX32664 - Biometric Sensor Hub 24
      4.4.5 Arduino as an Industrial Component 24
4.5 Development Kits (System-on-Chip and Microcontroller)
  4.5.1 Option 1:
    4.5.1.1 SparkFun Pro nRF52840 Mini - Bluetooth Development Board
  4.5.2 Option 2:
    4.5.2.1 Arduino Nano 33 BLE
  4.5.3 Option 3:
    4.5.3.1 Arduino Pro Mini 328 - 3.3V/8MHz
  4.5.4 Option 4:
    4.5.4.1 ATtiny85
  4.5.5 Bluetooth Communication
    4.5.5.1 Bluetooth SMD Module - RN-41 (v6.15)
    4.5.5.2 nRF52832 Bluetooth Low Energy Module - MDBT42Q-512KV2

4.6 Battery
  4.6.1 Lithium Ion Battery - 850mAh

4.7 Technical Specifications of the Prototype Option Designs

4.8 Implementation

4.9 Set up Arduino Core

4.10 Quantized Data Transfer
  4.10.1 Oxygen Measurement converted to binary example:
  4.10.2 How much data can we store in the RAM?
  4.10.3 Timestamp
  4.10.4 Checksum:
  4.10.5 Implementation:
  4.10.6 Range of Oximeter:

4.11 Optimizing Code
  4.11.1 Remove Serial Print lines
  4.11.2 Examine the “types” being used and minimize to smaller primitives
  4.11.3 Remove smart-timers and use smart delays

4.12 Power Profiling
  4.12.1 USB Power Meter

4.13 Profile (Current Consumption)

5. Results

6. Discussion
7. Conclusion 41
8. Future Work 42
9. References: 43
10. Appendix 46
List of Tables

Table 1: ZigBee comparison of pros and cons............................................................................................................. 20
Table 2: Bluetooth comparisons. ......................................................................................................................................... 21
Table 3: Comparison of Prototype Designs.................................................................................................................. 32
Table 4: Thresholds for measurements to be quantized............................................................................................. 34
Table 5: Trade-off - battery vs. memory storage (recording intervals). ........................................................................ 36
Table 6: Power Meters for power profiling.................................................................................................................. 39
Table 7: Steps taken to characterize the power profile of the nRF52840. ..................................................................... 40
List of Figures

Figure 1: Oxygen and Carbon Dioxide transport in the human body [8]. ................................................................. 3
Figure 2: Binding sites become progressively occupied until saturation [8]. .......................................................... 5
Figure 3: A schematic of pulsatile arterial blood and other blood and tissue components [25]. ............................... 8
Figure 4: The DC and AC components of the PPG signals for different LED wavelengths [25]. ............................. 9
Figure 5: Molar absorption coefficients of HbO2 and RHb [25]. .............................................................................. 9
Figure 6: Light attenuation through non-scattering media [9]. ............................................................................... 10
Figure 7: The total intensity of light is a linear superposition of each absorbent [9]. ............................................ 10
Figure 8: Light absorption of different molecules (RHb, HbO2, HbCO, HbMet) [25]. ............................................. 11
Figure 9: Light transmission during plethysmography measurement [9]. ................................................................. 13
Figure 10: Theoretical and calibrated R-curves. All devices (in this case, phones) use the same sensor with different optical shields. Notice the slightly different R-curves [25] .............................................................. 14
Figure 11: Example of R-curve based on the collected data from the calibration lab. Each color represents a different test subject [25]. ............................................................................................................... 15
Figure 12: Pulse oximeter design from TI................................................................................................................... 16
Figure 13: Pulse Oximeter from Arrow/Microchip with LCD, large batteries and extended transmissive measurement .......................................................... 18
Figure 14: Pulse Oximeter from Analog with wired power supply and LCD ............................................................. 19
Figure 15: Dimensions of the SparkFun Pulse Oximeter. ......................................................................................... 22
Figure 16: Voltage and ground connections of the SparkFun Pulse Oximeter ....................................................... 23
Figure 17: MFIO and Reset (not connected to the Qwicc connector and must be soldered) ................................. 23
Figure 18: Biometric sensor hub of the Sparkfun Pulse Oximeter .............................................................................. 24
Figure 19: Component options for the sensor prototype. ....................................................................................... 26
Figure 20: NRF52840 dimensions and board layout ................................................................................................. 27
Figure 21: Arduino Nano BLE board ....................................................................................................................... 28
Figure 22: Arduino Pro Mini 328 board ..................................................................................................................... 29
Figure 23: ATTiny85 Microchip microcontroller .................................................................................................... 29
Figure 24: Bluetooth module with serial data transfer ............................................................................................ 30
Figure 25: BLE Low Energy Module with a form-factor smaller than a quarter. .................................................... 30
Figure 26: Lithium Ion Battery offering 3.7V at various mAh ranges (110, 850, etc.) ............................................... 31
List of Equations

Equation 1: $\text{SpO}_2$ as an estimate of the amount of oxygen in capillary blood. ............................................................. 7
Equation 2: Beer-Lamber law describing attenuation of light. ............................................................................................. 7
Equation 3: An expansion of the Beer-Lambert law. ........................................................................................................... 7
Equation 4: Layers of absorbents in a sample. .................................................................................................................. 10
Equation 5: Absorbance of tissue components in a sample. ............................................................................................... 11
Equation 6: Differential absorption, the time-derivative of the total absorbance. .......................................................... 12
Equation 7: Assessing differential absorption through transmitted intensity. ................................................................. 12
Equation 8: Ratio of two differential absorptions (at different wavelengths) used to obtain $\text{SpO}_2$. ......................... 12
Equation 9: Assuming that the heartbeat arterial-layers are equal. ................................................................................... 13
Equation 10: Solve for $\text{SpO}_2$ using Ratio of Ratio equation. ......................................................................................... 13
Equation 11: Short-hand notation of the $\text{SpO}_2$ measurement formula. ......................................................................... 15
Equation 12: $R$ as found in Equation 10. ....................................................................................................................... 16
1. Introduction

More than 1.5 million adults in the United States use supplemental oxygen for a variety of respiratory disorders to improve their quality of life and prolong survival [1]. Our study will focus on Chronic Obstructive Pulmonary Disease (COPD) patients that monitor oxygen saturation using SPO₂ meters to detect potentially fatal oxygen levels. It is estimated that about 30 million people in the United States have COPD, a large percentage of which utilize oxygen therapy for treatment and prolonging survival.

Through oxygen therapy, supplemental oxygen provided through a mask or nasal cannula allows patients to breathe more easily and a portable oxygen tank makes it easier to get around. The oxygen tank has gauges and nozzles to increase or decrease oxygen intake (usually performed by a healthcare provider or guardian) based on a patient’s oxygen saturation – usually measured using pulse oximetry.

The principle of pulse oximetry revolves around the fact that the arterial component of blood is pulsatile in nature (time varying). When light is made incident on the human body, the amount of light that passes through after the attenuation from various components like tissue, artery and veins also has a pulsatile component riding over a constant component. The aim of pulse oximetry is to measure the percentage of oxygenated hemoglobin (HbO₂) to the total hemoglobin (Hb) in the arterial blood. This is referred to as SpO₂ monitoring [2].

Light measurements are performed at two wavelengths 650nm (Red) and 950 nm (infrared) that have different absorption coefficients, allowing SPO₂ to be calculated by cancellation of common components. Measurement might occur by transmission or reflection. In the transmissive case, the light source and photodiode are placed on opposite ends (usually, of the finger), with the photodiode collecting residual transmitted light. In the reflective case, the photodiode and light source are on the same side of the skin and measure the reflectivity of the gases in the blood. We will be focusing on the reflective method in this study.

The most common method to measure a patient’s oxygen saturation is using a pulse oximeter - a medical device lightly clipped onto the finger of the hand that sends light through the blood vessels of the finger and measures the amount of red and infrared light reflected off the gases. This device is uncomfortably heavy, large and requires manual operation.

Previous work has involved larger designs following a similar style to the classical oximeter. Charging mechanisms in place are not utilizing lithium-ion batteries and fail to provide a comfortable solution for hospital patients. Our goal is to design a pulse oximeter prototype that can be incorporated with a fashion fabric, allowing COPD patients convenience and comfort.
1.1 Goals of this Study
We will design a prototype of a lightweight oximeter on a flexible or small circuit board that will operate without a display and other current-inducing components. Our prototype will send data wirelessly and feature wireless-charging capabilities providing oxygen therapy patients an IoT enabled, flexible and lightweight device offering potential for automatic detection and agile response to oxygen saturation readings. Our key components will be an LED/photodiode sensor, AFE (analog front-end to process the result of sensor measurements and send to external processor/MCU), wireless charging battery and wireless module for sending and receiving data.

1.2 Background
Oxygen transport is necessary for living organs as oxygen is continuously needed for metabolic reactions that release energy from nutrient molecules and produce ATP. At the same time, these reactions release carbon dioxide (CO$_2$). Excessive amounts of CO$_2$ can produce acidity that can be toxic to cells so it must be removed quickly and efficiently. Together, the cardiovascular system and respiratory systems cooperate to supply O$_2$ and remove CO$_2$. The respiratory system provides for gas exchange or the intake of O$_2$ and the removal of CO$_2$ and the cardiovascular system transports blood containing the gases between the lungs and body’s cells. Failure of either of these two systems disrupts internal equilibrium (also known as homeostasis) and causes a rapid death of cells from oxygen starvation and buildup of waste products.

We will spend this chapter reviewing how oxygen reaches cells and then discuss how oxygen is transported to tissues through the circulatory and respiratory systems. Then we will elaborate on pulse oximetry as a common tool for blood oxygen saturation measurement, concluding with an introduction to the wearable pulse oximeter market.

1.3 The Circulatory System and Oxygen Transport
The circulatory system is designed to ensure the survival of all cells of the body at every moment. It achieves this by maintaining the immediate chemical environment of each cell in the body (i.e., the interstitial fluid) at a composition appropriate for that cell’s normal function. The term “homeostasis” is used to denote the approximate constancy of the internal environment (3). Often a limiting factor for cell survival is oxygen availability which is generally supplied to a cell through the process of passive diffusion. Krogh theorized that oxygen is transported in the tissue by passive diffusion driven by gradients of oxygen tension (PO$_2$) [4].

1.4 Tissue Oxygenation
The main purpose of the respiratory system is to deliver oxygen to tissue cells and to remove the carbon dioxide produced by them. Tissue oxygenation is usually described by one of the following three terms: Normoxia (normal oxygen levels), Hypoxia (deficiency of oxygen reaching tissues of the body) and Anoxia (absence of oxygen everywhere in the tissue).
Tissue oxygenation can be described as a multi-staged process where respiratory and cardiovascular systems cooperate to supply oxygen ($O_2$) and eliminate carbon dioxide ($CO_2$) in the body. The process shown in Figure 1 demonstrates how the respiratory systems handles gas exchange and the cardiovascular system handles oxygen and carbon dioxide transportation in the blood.

Under normal conditions in human circulation, each milliliter of blood carries about 0.2 ml of oxygen. In arterial blood, about 98% of this oxygen is reversibly bound to hemoglobin (a protein) contained within the red blood cells, and the remaining oxygen is in a free form, dissolved in both blood plasma and in the hemoglobin solution inside the red blood cells. Hemoglobin molecules in almost all vertebrates are tetramers, consisting of four peptide chains, each with an incorporated heme group. The oxygenation of hemoglobin (binding of oxygen and hemoglobin) takes place through $O_2$ binding to the Fe atom in each of the heme groups, preventing hemoglobin from binding to further oxygen or other molecules. The fraction
of available oxygen-binding sites occupied by oxygen is called hemoglobin-oxygen saturation, $S$ or $SO_2$; expressed as either a fraction or a percent [5].

For mammals, the bathing medium for cells is water and the total percentage of water compared to body weight is about 60%. For a 120 lb. person, water occupies about 72 pounds and is distributed among three compartments with the following approximate volumes: intracellular – 33% (39.6 lbs.), interstitial – 22.5% (27 lbs.), and circulating plasma – 4.5% (5.4 lbs.).

The various organs and tissues can be classified as one of two broad types: (1) blood “re-conditioners” and (2) “essential” tissues. The main purpose of the blood re-conditioners is to maintain the composition of the interstitial fluid relatively constant under all conditions experienced by the body. Flows to these tissues usually exceed metabolic needs. Examples of this type of tissue are the lung, which ensures proper exchange of oxygen and carbon dioxide; the kidney, which maintains electrolyte composition and fluid balance; the gut, which oversees nutrient absorption; and the skin, which is involved in temperature regulation.

The “essential” tissues are those whose function is always critical. The blood flows to these tissues typically match their metabolic needs. Examples of this type of tissue are the heart, which requires a continuous supply of energy to maintain its pumping activity, and the brain, which requires a continuous supply of nutrients and a need for the washout of metabolic products in order to maintain consciousness and carry out its critical functions. One can also add skeletal muscle during exercise to this list, since its energy requirements and needs for washout of metabolic products can be substantial.

As oxygen molecules diffuse into the cell, they are consumed, so that there is a progressive fall in oxygen concentration from the surface of the cell to the lowest concentration which occurs at the center of the cell [6].

1.5 The Respiratory System and Oxygen Transport
The protein hemoglobin is a molecule which is responsible for carrying almost all the oxygen in the blood. It is composed of four subunits, each with a heme group plus a globin chain [6]. The blood of a normal adult human contains at least six different species of hemoglobin molecules, all of which appear and function the same. Of these six or more species, hemoglobin A (or HbA) makes up about 92% of the total hemoglobin concentration in an adult human.

1.6 Binding of Oxygen to Hemoglobin: Oxygen Saturation Curve
The hemoglobin molecule has four binding sites for oxygen molecules which are the iron atoms in the four heme groups. This results in each Hb molecule binding with four oxygen molecules. The result of a sample of deoxygenated blood being equilibrated in steps with gas mixtures of increasing $PO_2$ (partial pressure of oxygen) is shown in Figure 2. The binding sites will become progressively occupied until, at a high-enough pressure of oxygen, all binding sites will be occupied. Factors including temperature, pH and the presence of carbon monoxide can all
cause the curve to shift to the right, adversely affecting the ability of blood to oxygenate. This has special importance during exercise and involuntary inhalation of carbon monoxide.

![Graph showing the oxygen saturation curve](image)

*Figure 2: Binding sites become progressively occupied until saturation [8].*

### 1.7 Gas Laws: Dalton’s Law and Henry’s Law

The exchange of oxygen and carbon dioxide between alveolar air and pulmonary blood occurs via passive diffusion which is governed by the behavior of gases as described by two gas laws, Dalton’s law, and Henry’s law. Dalton’s law describes how gases move down their pressure gradients by diffusion and Henry’s law explains the mechanism for how solubility of a gas relates to its diffusion. According to Dalton’s law, each gas in a mixture of gases exerts its own pressure as if no other gases were present.

Henry’s law dictates that the quantity of a gas that will dissolve in a liquid is proportional to the partial pressure of the gas and its solubility. In body fluids, the ability of a gas to stay in solution is greater when its partial pressure is higher and when it has a high solubility in water. In comparison to oxygen, much more CO₂ is dissolved in blood plasma because the solubility of CO₂ is 24 times greater than that of O₂.

#### 1.7.1 Exchange of Oxygen and Carbon Dioxide

In tissue oxygenation process, pulmonary ventilation (breathing) moves the ambient air in and out of the alveoli of the lungs through a series of cavities and tubes of the respiratory system. Air flows between the atmosphere and the lungs because of pressure differences caused by muscles in the respiratory system. Ventilation is regulated by neurons receiving data from mechanical and chemical receptors, acting as sensors. An example of this is the ventilation rate decreasing after exercise due to a return to normal carbon dioxide-oxygen concentrations in
the circulatory system. Deoxygenated exits the right ventricle and flows into the lungs to become reoxygenated. It leaves the lung by the pulmonary vein and enters the left atrium, left ventricle and then into the aorta, eventually reaching all extremities of the body.

2. Pulse Oximetry
Pulse oximetry (percentage of SpO\textsubscript{2} concentration in blood) has been used as a key health indicator for many decades. Although the original academic development was made in 1935, the modern basis for determining the SpO\textsubscript{2} concentration using light sources and photosensor(s) was developed by Takuo Aoyagi and Michio Kishi in 1972. When commercially feasible, SpO\textsubscript{2} concentration measurement devices have made huge gains in medical applications. Since 1987, the Standard of Care (SoC) for the administration of a general anesthetics has included pulse oximetry. All modern hospital bedside equipment include an SpO\textsubscript{2} module based on the same fundamentals, albeit with minor modifications.

These hospital-based devices are expensive and bulky, and in their current form, their use is limited to hospitals, health clinics, and some doctor’s offices. To enable individuals interested in tracking their body’s key health indicators, a solution that is both light enough to wear in comfort and cheap enough for a typical consumer to purchase is needed.\cite{7}

In addition to heart rate, blood pressure, respiratory rate, and temperature, pulse oximetry (PO) is considered to be the “fifth vital sign” of health status. Hemoglobin (Hb) is an essential part of the red blood cells responsible for delivery of oxygen from the lungs to the tissues. The amount of oxygen (O\textsuperscript{2}) bound to Hb at any time is called oxygen saturation. Expressed as a percentage, the oxygen saturation (SpO\textsubscript{2}) is the ratio of the amount of oxygen bound to the Hb to the oxygen carrying capacity of the Hb. Pulse oximetry provides a noninvasive way to measure the SpO\textsubscript{2}, or arterial hemoglobin saturation. Pulse oximetry also detects arterial blood pulsations, and therefore also calculates and reports a patient’s heart rate. A pulse oximeter is a medical device that measures the amount of oxygen in a patient’s arterial blood.

A typical oximetry sensor has a pair of light-emitting diodes (LEDs) facing a photodiode through a translucent part of the patient’s body, usually a fingertip or an earlobe. One LED is red, with wavelength of 660 nm; the other is infrared, with a wavelength of 940 nm. The percentage of blood oxygen is calculated based on the absorption rate from each wavelength of light after it passes through the patient’s body.\cite{7}

2.2 Principles of SpO\textsubscript{2} Measurement
We now discuss SpO\textsubscript{2} measurements and describe the process in which it is obtained from patients.

2.2.1 What is SpO\textsubscript{2}?
Hemoglobin (Hb) is an oxygen-transport protein in red blood cells (RBCs). The two main forms of Hb present in blood are oxygenated hemoglobin (oxy-hemoglobin, HbO\textsubscript{2}) and deoxygenated hemoglobin (deoxygenated-hemoglobin, RHb).
SpO₂ is a measure of the peripheral capillary oxygen saturation. More specifically, SpO₂ is an estimate of the amount of oxygen in capillary blood, which is described as a percentage of the amount of oxyhemoglobin to total hemoglobin, expressed as follows:

$$\text{SpO}_2 = 100 \times \frac{\text{C[HbO}_2]}{\text{C[HbO}_2] + \text{C[RHb]}}$$

*Equation 1: SpO₂ as an estimate of the amount of oxygen in capillary blood.*

where \(\text{C[HbO}_2]\) and \(\text{C[RHb]}\) are the concentrations of HbO₂ and RHb, respectively.

### 2.2.2 Beer-Lambert Law

The Beer–Lambert law describes the attenuation of light with the properties of the material through which the light is travelling. According to the Beer-Lambert law,

$$A = \ln \left(\frac{I_0}{I}\right) = \varepsilon(\lambda)Cd$$

*Equation 2: Beer-Lamber law describing attenuation of light.*

where \(A\) is the attenuation, \(I_0\) is the incident light intensity, \(I\) is the received light intensity, \(\varepsilon(\lambda)\) is the molar extinction coefficient, \(C\) is the concentration of material, and \(d\) is the optical path length. Considering the molecule compound of tissue, Beer-Lambert law can be extended as follows:

$$A = d \left[ \varepsilon_{\text{HbO}_2}(\lambda)C[\text{HbO}_2] + \varepsilon_{\text{RHb}}(\lambda)C[\text{RHb}] + \varepsilon_{\text{other}}(\lambda)C[\text{other}] \right]$$

*Equation 3: An expansion of the Beer-Lambert law.*

The Beer-Lambert law enables us to measure SpO₂ by using the molar extinction coefficients of HbO₂ and RHb [8].

### 2.2.3 Pulse Oximetry

Pulse oximetry is a tool used for the noninvasive measurement of blood oxygenation (i.e., SpO₂). Pulse oximetry is based on two principles: modulation of transmitted light by absorption of pulsatile arterial blood and different absorption characteristics of HbO₂ and RHb for different wavelengths.

Pulse oximetry can be classified as transmissive and reflective: Transmissive pulse oximetry is when the photodiode and the LED are placed on opposite sides of the human body (i.e. finger).
The body tissue absorbs some of the light, and the photodiode collects the residual light that passes through the body. Reflective pulse oximetry is when the photodiode and the LED are on the same side. The photodiode collects the light reflected from various depths underneath the skin. Recent developments have caused both methods to have similar levels of accuracy with transmissive pulse oximetry performing more reliably. Figure 3 shows pulsatile arterial blood and other blood and tissue components that affect transmission of light.

The pulsatile arterial blood absorbs and modulates the incident light passing through the tissue and forms the photoplethysmography (PPG) signal, as shown in Figure 4. The AC component of the PPG signals represents the light absorbed by the pulsatile arterial blood. This AC component is superimposed on a DC signal that captures the effects of light absorbed by other blood and tissue components (e.g. venous and capillary blood, bone, water, etc.).

Note that the DC and AC components of the received PPG signals are different for different LED wavelengths. This is due to the different absorption characteristics of HbO₂, RHb, and other tissue components for different wavelengths.
Figure 4: The DC and AC components of the PPG signals for different LED wavelengths [25].

Figure 5 shows the molar absorption coefficients of HbO$_2$ and RHb. To measure SpO$_2$, two LEDs with different wavelengths are required. In addition, these two wavelengths should be selected such that the molar absorption coefficients of HbO$_2$ and RHb are well separated. A red LED at 660nm and an infrared LED at 880nm are commonly used in pulse oximetry [8].

![Figure 5: Molar absorption coefficients of HbO$_2$ and RHb [25].](image)

2.3 Beer-Lambert Law in Oximetry
The measurement principle of oximetry is based on Beer-Lambert law presented in Figure 6 that describes light attenuation through a sample of homogenous non-scattering media.
Mathematically the law is expressed as $I = I_0 e^{-\varepsilon \lambda cl}$, (3) where $I$ is the transmitted intensity of the light, $I_0$ is the incident intensity, $\varepsilon \lambda$ is the extinction coefficient or absorptivity of the absorbent at wavelength $\lambda$, $c$ is the concentration of the absorbent, and $l$ the optical path-length of the sample. Beer-Lambert law is valid when there is one unknown substance in clear, non-turbid solution and constant path-length [9]. In addition, there should be no chemical reactions or reactions between absorbent and solvent. In cases where the sample consists of layers of absorbents as presented in Figure 7, the total intensity of transmitted light presented in Equation 4 is a linear superposition of intensities of each absorbent.

\[ I = I_0 e^{\sum -\varepsilon \lambda, a c a l a} = I_0 e^{-A} \]

*Figure 6: Light attenuation through non-scattering media [9].*

*Figure 7: The total intensity of light is a linear superposition of each absorbent [9].*

where ‘a’ represents an absorbent or analyte. Oximetry is based on the change in the absorption of electromagnetic energy (color) of hemoglobin molecule when its chemical binding is altered. Oxyhemoglobin, reduced hemoglobin, carboxyhemoglobin, and methemoglobin have their own characteristic colors that can be described with extinction
coefficients. The extinction coefficients of four most common hemoglobin species are presented in Figure 8 in the wavelength range from 570 to 1000 nm the visible light end of the spectrum into the near infrared. Figure 8 shows that HbCO is not absorbing almost at all at long wavelength and has quite similar absorption with oxyhemoglobin with short wavelengths. Methemoglobin absorbs heavily around 640 nm and wavelengths above 840 nm. Wavelengths most used in pulse oximetry 660 nm (red) and 940 nm (infrared) are marked with vertical lines. Their selection is based on large difference in absorptions of HbO2 and RHb, manageable tissue absorption and flatness of spectrum [10, 11].

![Figure 8: Light absorption of different molecules (RHb, HbO\textsubscript{2}, HbCO, HbMet) [25].](image)

2.4 Theory of Pulse Oximetry
Absorbance (A) at wavelength \( \lambda \) is defined as the negative natural logarithm of the fraction of light that passes through a sample called transmission \( T \). Combined with Equation 4 the absorbance can be written in form

\[
A_\lambda = -\ln(T) = -\ln\left(\frac{I}{I_0}\right) = \sum_a \varepsilon_{\lambda,a} \cdot c_a \cdot l_a,
\]

*Equation 5: Absorbance of tissue components in a sample.*

where \( a \) represents all tissue components such as arterial blood, venous blood, skin, and bones in the sample. Figure 9 presents the how the photoplethysmography (PPG) signal is formed. The lower part presents the tissue layers in a sample and upper part shows the transmission signal. DC component of transmission signal is defined as the transmission without blood volume pulsation. DC component varies only slowly due to respiration, sympathetic nervous system
activity, thermo-regulation, and changes in total hemoglobin concentration. In pulse oximeters, the time-dependent arterial pulsation of transmission signal, also called AC component, is separated from the total transmission signal. The strength of the AC component is only about one to two percent of the total transmission. It is the time-derivative of the total absorbance where the only time-dependent signal is the path length, $l_{\text{art}}$, through arterial blood. Differential absorption ($d\Delta\lambda$) is presented in Equation 6.

$$d\Delta\lambda = \frac{d(A\lambda)}{dt} \cdot \Delta t = \frac{dl_{\text{art}}}{dt} \cdot \sum \varepsilon_{\lambda,\text{art}} \cdot c_{\text{art}} \cdot \Delta t.$$  

*Equation 6: Differential absorption, the time-derivative of the total absorbance.*

On the other hand, differential absorption can be assessed through transmitted intensity (Equation 7) through the arterial blood component as written in equation

$$d\Delta\lambda = \frac{d \left( -\ln \left( \frac{I(t)}{I_0} \right) \right)}{dt} \cdot \Delta t = \frac{I'(t)}{I(t)} \cdot \Delta t \approx \frac{AC\lambda}{DC\lambda} = \frac{I_{\text{max}} - I_{\text{min}}}{I_{\text{max}}},$$

*Equation 7: Assessing differential absorption through transmitted intensity.*

During one pulse differential absorption can be approximated with the DC component ($DC\lambda$) and AC component ($AC\lambda$) intensities, that can be calculated from the minimum transmission after systolic rise, $I_{\text{min}}$, and diastolic maximum transmission of light, $I_{\text{max}}$. Current conventional two wavelength pulse oximeters use light-emitting diodes (LEDs) of wavelengths 660 nm (red light) and 940 nm (near-infrared light) and semiconductor photodetector to obtain plethysmography signals from an extremity such as finger or earlobe. Ratio of two differential absorptions with different wavelengths, ratio-of-ratios ($R$), is calculated to obtain $\text{SpO}_2$. Based on theory, $R$ can be presented with equation

$$R = \frac{dA_{\text{red}}}{dA_{\text{ired}}} = \frac{(\varepsilon_{\text{HbO}_2,\text{red}} \cdot c_{\text{HbO}_2} + \varepsilon_{\text{HHb,red}} \cdot c_{\text{HHb}}) \cdot \Delta l}{(\varepsilon_{\text{HbO}_2,\text{ired}} \cdot c_{\text{HbO}_2} + \varepsilon_{\text{HHb,ired}} \cdot c_{\text{HHb}}) \cdot \Delta l},$$

*Equation 8: Ratio of two differential absorptions (at different wavelengths) used to obtain $\text{SpO}_2$.*

When the thickness of the heartbeat-added arterial layer, $\Delta l$, is estimated to be equal for both wavelengths, assumption, $t\text{Hb} \cdot \text{SpO}_2 \approx c\text{HbO}_2$, is made, and
Figure 9: Light transmission during plethysmography measurement [9].

Constraints of two hemoglobin species, cRhb + cHbO2 = tHb, is included, the equation can be rewritten in form

\[
R = \frac{\varepsilon_{HbO2,red} \cdot SpO2 + \varepsilon_{Rhb,red} \cdot (1 - SpO2)}{\varepsilon_{HbO2,red} \cdot SpO2 + \varepsilon_{Rhb,red} \cdot (1 - SpO2)}.
\]

Equation 9: Assuming that the heartbeat arterial-layers are equal.

which can be solved for SpO2 as presented in equation

\[
SpO2 = \frac{\varepsilon_{Rhb,red} - R \cdot \varepsilon_{Rhb,red}}{(\varepsilon_{HbO2,red} - \varepsilon_{Rhb,red}) - R \cdot (\varepsilon_{HbO2,red} - \varepsilon_{Rhb,red})}.
\]

Equation 10: Solve for SpO2 using Ratio of Ratio equation.

Theory presented above is based on the Beer-Lambert assumptions that, in reality, are not valid in in-vivo tissues. Other substances are present in blood and the incident light is partially scattered, refracted and reflected. The detected light at the photodetector consists of photons that travel different routes. Some of the photons travel through the tissue without migrating from the line that emitter and detector form and some scatter farther from the line and still reach the detector at the other end. When the absorption increases the detected light tends to come more from the shorter paths. Because of this path lengths vary with different
wavelengths. Because of these deviations from the Beer-Lambert law conventional pulse oximeters are empirically calibrated to give better estimation of oxygen saturation. [10, 12]

2.5 Why is Calibration Required?

According to these regulations, manufacturers need to declare the calibration range, reference, accuracy, methods of calibration and range of displayed saturation level. Furthermore, for the performance assessment, the FDA requires at least 200 data points equally spaced over a saturation range of 70% to 100%. Test subjects should have different ages, gender, and skin tones. For instance, the FDA requires that at least 30% of the volunteers must have dark skin pigmentation. The overall error or the root mean square error (RMSE) must be below 3.0% for transmissive pulse oximetry and below 3.5% for reflective pulse oximetry.

In addition to the FDA regulations, the theoretical relation between R and SpO2 does not give a satisfactory SpO2 measurement accuracy due to ideal-case assumptions used in pulse oximetry and the different optical properties of the hardware (e.g., optical shield and wide light spectrum of LEDs). Figure 10 shows the theoretical R-curve and calibrated R-curves. The theoretical R-curve does not give accurate SpO2 values.

![Figure 10: Theoretical and calibrated R-curves. All devices (in this case, phones) use the same sensor with different optical shields. Notice the slightly different R-curves [25].](image)
In addition, Figure 4 shows the effect of a cover shield in front of the sensor. Although, all phones use the same sensor, their calibrated R-curves are slightly different from each other. Thus, a calibration process is required to obtain calibration coefficients for better measurement accuracy by compensating for the deviations from the Beer-Lambert law and the non-idealities of the hardware. These coefficients are obtained after collecting comprehensive data in a 3rd party calibration lab.

In the calibration lab, the SpO\textsubscript{2} level of the test subjects are varied in a controlled manner and the PPG signals measured from the test subjects are recorded. During data collection, the test subjects use a gas mask to control their SpO\textsubscript{2} level. Through the gas mask, the blood oxygen content is reduced incrementally by changing the oxygen level of the test subject from 100% SpO\textsubscript{2} and lowering to 70% SpO\textsubscript{2}. After collecting enough data according to the FDA regulations, the recorded PPG signals are used to find the R values. Then, a second (or first) order line is fitted to obtain the calibration coefficients a, b, and c for the SpO\textsubscript{2} measurement algorithm, as shown in Figure 11 [6, 13].

![Figure 11: Example of R-curve based on the collected data from the calibration lab. Each color represents a different test subject.](image)

SpO\textsubscript{2} measurement is achieved by the following equation:

\[
\text{SpO}_2 = aR^2 + bR + c
\]

*Equation 11: Short-hand notation of the SpO\textsubscript{2} measurement formula.*

where R is determined by the following equation:
and \( a, b, \) and \( c \) are calibration coefficients.

3. Related Work

Now we will explore related work that have sought to produce a design with a small form-factor. We will describe each of the different designs and assess the viability of each based on metrics such as: wearability, power consumption, form-factor, and scalability.

3.1 Pulse Oximeter from Texas Instruments

Figure 12 is a reference design for a miniaturized pulse oximeter specifically targeted to high end clinical applications. The small form factor module can simplify and accelerate pulse oximeter reading and prescription. It features Texas Instrument’s AFE4403 Analog Front End for interfacing with the LED and Photodiode sensors, the design also includes an MCU for processing the information from the AFE.

The AFE4403 is a fully integrated analog front-end (AFE) ideally suited for pulse oximeter applications. It consists of a low-noise receiver channel with an integrated analog-to-digital converter (ADC), an LED transmit section along with diagnostics for sensor and LED fault detection. The device is a very configurable timing controller resulting in flexibility that enables

\[
R = \frac{AC_{\text{red}}/DC_{\text{red}}}{AC_{\text{ired}}/DC_{\text{ired}}}
\]

*Equation 12: R as found in Equation 10.*
the user to have complete control of the device’s timing characteristics. To ease clocking requirements and provide a low-jitter clock to the AFE4403, an oscillator is also integrated that functions from an external crystal. The device communicates to an external microcontroller using an SPI interface.

The TI MSP430 family of ultra-low-power microcontrollers consists of several devices featuring peripheral sets targeted for a variety of applications. The architecture, combined with extensive low-power modes, is optimized to achieve extended battery life in portable measurement applications. The microcontroller features a powerful 16-bit RISC CPU, 16-bit registers, and constant generators that contribute to maximum code efficiency. The digitally controlled oscillator (DCO) allows the devices to wake up from low-power modes to active mode in 3.5 µs (typical) [2].

Code Composer Studio is an integrated development environment (IDE) that supports TI's Microcontroller and Embedded Processors. Code Composer Studio contains a set of tools used to develop and debug embedded applications. It includes a C/C++ compiler, source code editor, project build environment, debugger, profiler, and many other features. The IDE also provides a single user interface aiding in application development. Code Composer Studio combines the Eclipse software framework with embedded debug capabilities from TI [15].

This design supports oxygen-saturation measurement via an extremity (finger, toe) but is not fine-tuned. It features an AFE4403 and third-party optical sensor module for measuring oxygen saturation and MSP430F5528 MCU for running algorithms to process each measurement. This design is tested but incomplete and requires further development of schematics, layout, Gerber files, and an improved BOM. The design is not suitable for the product development environment of our team. Since we are experimenting with advanced materials, a more abstracted design would be more suitable.

3.2 Pulse Oximeter from Arrow (and Microchip Technologies)

Arrow and Microchip offer the dsPIC33FJ128GP Family Pulse Oximeter (Figure 13) providing heart rate and blood oxygen level measurement that can be taken using wired transmissive pulse oximeter sensors and by wrist-worn fitness activity trackers. The Pulse Oximeter Demonstration Board was implemented using the dsPIC33FJ128GP802 Digital Signal Controller [16].

This design requires significant development effort on the algorithm and board layout aspects. It also does not boast a strong algorithm and processing backbone that is required along the product roadmap for FDA approval and implementation in a clinical environment. Furthermore, it utilizes an on-board display and lacks wireless capabilities (both of which must be developed and included) and will not be suitable for our design.
From a hardware and software perspective, we see that the design utilizes two triple-A batteries (1000 mAH each) and a current-inducing display that must be deactivated and repurposed.

![Figure 13: Pulse Oximeter from Arrow/Microchip with LCD, large batteries and extended transmissive measurement.](image)

### 3.3 Pulse Oximeter from Analog

The Precision Analog Microcontroller products shown in Figure 14 from Analog Devices includes the key analog building blocks required by a high-end oximetry design. The ADuC7024, used here, includes a high-performance multichannel 12-bit along with two 1-MSPS ADC and two DACs. These features allow recovery of the weak arterial pulsatile signals generally seen during low peripheral blood circulation [7].
The Micro Converter also includes a 32-bit ARM7TDMI core. It runs at an average of 41.8 MHz and provides an expandable processing platform. A proprietary digital signal processing algorithm, developed by ITEC Engineering, detects arterial blood pulsations, and detects and rejects motion artifacts, calculates SpO2 and heart rate values, and filters and scales the real time arterial blood pulsatile waveform.

The LEDs are powered sequentially through a MOSFET bridge. The photodiode amplifier converts received light into a voltage that is inversely proportional to the light absorbed by the patient’s tissue. The final measurement is made on red and infrared wavelengths alternatively.

The design contains an LCD display and has features for generating audio pulse tones via the onboard PWM. It is bulky and has a liquid LCD, speakers and other components that are undesirable for a low form-factor chip intended for wearable applications. It also has no mention of FDA approval aspirations and stops short of a viable algorithmic foundation. Instead, the device is packed full of components and must be tailored to the required use-case.

4. Product Management (Design)
After exploring previous efforts to design an ultra-low power and small form-factor pulse oximeter, we must now introduce the chosen sensor, explore wireless communication protocols and compare various System-on-Chip (SoC) microcontrollers that will allow us to maintain a clear trajectory to market, easy replicability (across long-distance teams) while maintaining low-power and form-factor at the heart of our design strategy.
4.1 Wireless Communication
We need to design a wireless communication component that will enable our product prototype to be IoT-enabled. Embedded systems can use a range of wireless technologies that have been designed for both telecom mobile devices and electronics (i.e. Bluetooth is used on mobile phones and found as a component in some embedded systems) A list of relevant technologies follows:

- ZigBee (IEEE 802.15.4)
- Bluetooth (IEEE 802.15.1)
- Wi-Fi (IEEE 802.11)
- IrDA
- RFID
- GSM, CDMA (Cellular)
- LoraWAN

For our study, we will only examine ZigBee and Bluetooth as they satisfy the basic requirements of our prototype – namely low-power, full-duplex communication and low-cost. Wi-Fi is disqualified due to high power requirements, RFID and IrDA are one-way (not full duplex), GSM is based on cellular networks and is an excessive range for our design which is expected to be hand-held and closely tied to mobile phones in nature.

4.2 ZigBee (IEEE 802.15.4)
The main field of application of ZigBee is between self-contained devices and equipment. It is an open wireless communication global standard designed to address the needs of low-power and low-cost wireless IoT networks. The main purpose of the ZigBee protocol is for home automation devices [17]. The ZigBee (IEEE 802.15.4) standard specifies the parameters of the physical layer (PHY) and medium access control (MAC), offering support for different network topologies (or structures) [18].

It mainly provides a short-medium communications range (about 10 meters) and a bandwidth of up to 250 kbit/s. At this rate, data transfer occurs within 2.4 GHz frequency range. Low-power consumption and data rate are due to a low-power coupling device. Using the frequencies associated with ZigBee are license-free. Table 1 shows pros and cons of ZigBee.

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low energy</td>
<td>Complex</td>
</tr>
<tr>
<td>Interoperability</td>
<td>Low-data transmission rate</td>
</tr>
<tr>
<td>Lower cost of devices (open source)</td>
<td>Not great for outdoor communication</td>
</tr>
<tr>
<td>Minimal team member experience</td>
<td>Lack of total solution</td>
</tr>
</tbody>
</table>

Table 1: ZigBee comparison of pros and cons.
4.3 Bluetooth (IEEE 802.15.1)

Bluetooth is widely used in mobile handheld devices for information exchange within a radius of 10 to 100 meters: cell phones, headsets, wireless manipulators, and keyboards. Apart from extensive uses in consumer electronics, Bluetooth is also utilized in embedded systems for industrial use. An example would be systems for local monitoring of remote objects (basic cell stations and electrical substations).

Bluetooth devices are divided into three classes based on radiation power (under lab conditions):

- Class 1 – power of up to 100 mW, range up to 100m
- Class 2 – power of up to 2.5 mW, range up to 10 m
- Class 3 – power of up to 1 mW, range of up to 1 m

There are several Bluetooth specifications: v1.0, v1.1, v1.2, v2.0, v2.1, v3.0, v4.0, etc. The latest version is v5.0 (adopted in 2020). Data transfer speeds with Bluetooth 5 devices reach up to 2 Mbps (double of the last standard – 4.2) and can communicate over distances up to 800 feet (240 meters), which is 4x what is allowed by 4.2. Table 2 describes benefits of Bluetooth 5.

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep functionality</td>
<td></td>
</tr>
<tr>
<td>Range dependent on power supplied</td>
<td>Bluetooth security is weak compared to Wi-Fi and other standards</td>
</tr>
<tr>
<td>Low-power, energy efficient</td>
<td>Small range</td>
</tr>
<tr>
<td>High-speed (compared to ZigBee)</td>
<td>Compatibility</td>
</tr>
<tr>
<td>Easy use</td>
<td></td>
</tr>
<tr>
<td>Extensive team member experience</td>
<td></td>
</tr>
</tbody>
</table>

| Table 2: Bluetooth comparisons. |

Bluetooth is more well-suited to our prototype because it fits our range and power requirements. The range is appropriate for our use-case (short-range – sensor on user’s hand to a mobile device), high data-rate to track heart rate and pulse oximetry data, sleep functionality and easy interoperability across devices [19]. This contrasts with ZigBee which is designed to carry small amounts of data over medium distances. It is also based on mesh topology – transmitter to gateway information transfer. It is a Local Area Network (LAN – like Wi-Fi), designed to connect to devices that need a wider range. In the end, both networks are complementary to each other, at times BLE is useful for IoT applications and other times, ZigBee is superior.

4.4 Sparkfun Pulse Oximeter and Heart Rate Sensor

The SparkFun Pulse Oximeter and Heart Rate Sensor is an I2C (address: 0×55) based biometric sensor, utilizing two chips from Maxim Integrated: the MAX32664 Biometric Sensor Hub and
the MAX30101 Pulse Oximetry and Heart Rate Module. The MAX32664 is a small Cortex M4 processor that handles all algorithmic calculations, filtering, pressure/position compensation, R-wave detection, and gain control. The Qwiic connector easily connects to the I²C data lines along with two additional lines that must be soldered (Reset and MFIO). The board (as seen in Figure 15) measures at 1in x 0.5in, easily fitting on the finger of the hand.

The MAX30101 performs sensing by utilizing internal LEDs for reflective oximetry. The resultant data is passed onto and analyzed by the MAX32664 which applies its algorithms to determine heart rate and blood oxygen saturation (SpO₂). SpO₂ results are reported as the percentage of hemoglobin that is saturated with oxygen. It also provides useful information such as the sensor's confidence in its reporting as well as a finger detection status data point. This design places heavy emphasis on its algorithmic backing and boasts an extremely small ready-to-go packaging. It is superior to other designs for its flexibility (it contains only necessary components).

The board also offers ultra-low power operation for mobile applications, fast output data capability (for a shorter “active” time) and offers confidence levels for measurements. Additionally, raw, and processed data is available, and the device can be calibrated to specific uses through on-board registers. It is superior to other designs in its flexibility, scalability, algorithmic emphasis (and future potential for FDA approval), connectivity (Qwiic connector allows easy prototyping with long-distance and remote teams) and low-power design. We will utilize this device for the rest of our project [20].

![Figure 15: Dimensions of the SparkFun Pulse Oximeter.](image)

4.4.1 Power and I²C Pins:
Power is provided through a 3.3V pin (that is also connected via Qwiic connector) and the I2C pins (SCL and SDA) are also accessible but can be utilized through the Qwiic connector. Figure 16 shows the “MAX32664 Side” convenient connection to the Qwiic connector that enables streamlining development efforts.
Figure 16: Voltage and ground connections of the SparkFun Pulse Oximeter.

4.4.2 Additional Pins:
This board has two additional pins on its header: the RESET and MFIO pin. These pins are required for the board to function because they determine if the board enters data collection mode or not.

Figure 17: MFIO and Reset (not connected to the Qwicc connector and must be soldered).

4.4.3 MAX30101 - Pulse Oximeter and Heart Rate Monitor
The SparkFun Pulse Oximeter works by placing the finger gently on the sensor in which it shines red, infrared, and sometimes green light through the skin. The capillaries filled with blood under the skin will absorb this light and the sensor will read which light comes back. This light data is then sent back to the Biometric Sensor Hub which handles all the calculations to determine heart rate and blood oxygen levels. The MAX30101 is shown in Figure 18.
4.4.4 MAX32664 - Biometric Sensor Hub
The MAX32664 Biometric Sensor Hub is a Cortex M4 micro-controller dedicated to receiving the data it receives from the MAX30101 and running the calculations to determine heart rate and blood oxygen. There are a multitude of settings to tailor the sensor through the Arduino Library which will utilized during the FDA approval process [21].

4.4.5 Arduino as an Industrial Component
For the industrial industry, the most common computing devices to control automation and other factors are PLCs (Programmable Logic Controllers). These are usually heavy duty and are made for use in manufacturing plants. An Arduino Uno cannot easily replace the PLC that most manufacturing process centers use. There is a reason why PLCs (which can be hundreds of dollars) are more expensive than Arduino boards.

PLCs are different from other computing devices as they are intended for severe conditions found in manufacturing plants. This means they can handle dust, higher and lower temperature, and moisture—environmental conditions the Arduino is not built to handle. PLCs also have more extensive input/outputs (I/O) to connect to other sensors and actuators. A PLC can output to other elements, including electric motors, magnetic relays, sirens, indicator lamps, and much more. This is something that the Arduino can also do, but it is more limited, as its analog inputs are only ranged from 0-5 V and the analog outputs are pulse-width modulation (PWM) based.

The evaluation hardware and development boards consist a microcontroller with 3V3 or 5V GPIOs that are directly forwarded to the connectors for user experiments. On industrial controllers you can also find the same or similar microcontrollers but their GPIOs are never directly accessible. The GPIOs are always equipped with level conversion (like 24V) and with several protection circuits. For inputs you normally have over-voltage protection, filters, and circuits preventing damage that could result from surges or bursts. For outputs you have current amplification to drive currents worth several Amperes and protection against overload and short circuit. All this helps the microcontroller to survive in a harsh industrial environment.

Another difference is that industrial controllers are tested against several industry standards (IEC, IEEE) for electrical safety in different environmental conditions and resistance to
mechanical shock and vibrations. One could build an industrial controller using an Arduino, if it is equipped with a "shield" as well as a mechanical enclosure that are appropriate to meet the above requirements, and to pass the industrial standard testing.

4.5 Development Kits (System-on-Chip and Microcontroller)
The options presented in Figure 19 are assuming that the front-end sensor is the SparkFun Pulse Oximeter and Heart Rate Sensor that sends data using I²C protocol. Design considerations include Bluetooth capability, low-power, form-factor, and as close to a plug-and-play solution as possible (for effective prototyping).
Figure 19: Component options for the sensor prototype.
4.5.1 Option 1:

4.5.1.1 SparkFun Pro nRF52840 Mini - Bluetooth Development Board
The SparkFun Pro nRF52840 Mini is a breakout and development board for Nordic Semiconductor’s nRF52840 – a combination of ARM Cortex-M4 CPU and 2.4GHz Bluetooth radio. The nRF52840 breaks out critical I/O pins including GPIO and those needed for power while maintaining a small form-factor that nearly matches that of the Arduino Pro Mini (except with the bonus of a Qwiic Connector). It features a USB interface (using the nRF52840’s native USB support), which can be used to program, power, and communicate with the chip making it able to be used for any data-communications purpose: UART, I²C, SPI. The Pro nRF52840 Mini features a Raytac MDBT50Q-P1M module. This module connects the nRF52840 to a trace antenna, fits the IC into an FCC-approved footprint, and includes a lot of the decoupling and timing mechanisms that would otherwise be required for a bare nRF52840 design. Also included onboard is a LiPo battery charger, a Qwiic connector, an on/off switch, a reset switch, and a user LED/button [22].

The board comes pre-programmed with a USB bootloader. Programs can be developed for the nRF52840’s Cortex-M4 either Arduino, Circuit Python, or C (using Nordic’s nRF5 SDK), and loaded using a USB serial or mass-storage interface. Figure 20 shows the dimensions and board layout of the nRF52840.

![Figure 20: NRF52840 dimensions and board layout.](image)

4.5.2 Option 2:

4.5.2.1 Arduino Nano 33 BLE
The Nano 33 BLE is designed for short range BT interactions and power conscious prototypes. It is built around the NINA B306 module, based on Nordic NRF52840, and contains a powerful Cortex M4F. Its architecture, fully compatible with Arduino IDE Online and Offline, has a 9-axis
Inertial Measurement Unit (IMU) and a reduced power consumption compared to other same size boards [10]. This allows the design of wearable devices and movement sensing projects that need to communicate to other devices at a close range. Arduino NANO 33 BLE is also ideal for automation projects thanks to the multiprotocol BT 5.0 radio [23].

4.5.3 Option 3:

4.5.3.1 Arduino Pro Mini 328 - 3.3V/8MHz

The Arduino Pro Mini is a 3.3V Arduino running the 8MHz bootloader. Arduino Pro Mini does not come with connectors populated allowing flexibility for connections. The Arduino Pro series is meant for advanced designers that understand concepts like system voltage (3.3V), connectors, and USB off-board bootloaders.

In order to accomplish a low-cost board, SMD components and a two-layer design were utilized. This board connects directly to the FTDI Basic Breakout board and supports auto-reset. The Arduino Pro Mini also works with the FTDI cable but the FTDI cable does not bring out the DTR pin so it has no auto-reset feature. There is a voltage regulator on board so it can accept voltage up to 12VDC [24]. This board also requires a separate Bluetooth module.
4.5.4 Option 4:

4.5.4.1 ATtiny85

The high-performance, low-power Microchip 8-bit AVR RISC-based microcontroller combines 8KB ISP flash memory, 512B EEPROM, 512-Byte SRAM, 6 general purpose I/O lines, 32 general purpose working registers, one 8-bit timer/counter with compare modes, one 8-bit high speed timer/counter, USI, internal and external Interrupts, 4-channel 10-bit A/D converter, programmable watchdog timer with internal oscillator, three software selectable power saving modes, and debugWIRE for on-chip debugging. The device achieves a throughput of 20 MIPS at 20 MHz and operates between 2.7-5.5 volts [26].

By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

This product is recommended for safety critical applications targeting both industrial and automotive products (IEC 61508 and ISO 26262). This device requires significant development effort on the hardware side. It also requires a Bluetooth module.
4.5.5 Bluetooth Communication

4.5.5.1 Bluetooth SMD Module - RN-41 (v6.15)
This module from Roving Networks has a low form-factor and sends and receives serial data. This Bluetooth module is designed to replace serial cables. The Bluetooth stack is completely encapsulated. The end user sees serial characters being transmitted back and forth. For example, if a user presses the 'A' character from a terminal program from a computer, an 'A' will be pushed out the TX pin of the Bluetooth module [27].

![Bluetooth module with serial data transfer.](image)

4.5.5.2 nRF52832 Bluetooth Low Energy Module - MDBT42Q-512KV2
This ultra-compact module shown in Figure 26 (smaller than a quarter) comes integrated with a BLE radio as well as most of the supporting circuity. However, it is a blank chip and must be programmed with a J-Link programmer [28]. Firmware is not provided with this module. For the purposes of our design, this requires soldering and advanced knowledge of circuitry that will delay product-to-market speed and is not ideal for prototyping.

![BLE Low Energy Module with a form-factor smaller than a quarter.](image)
4.6 Battery

4.6.1 Lithium Ion Battery - 850mAh

These are very slim, extremely lightweight batteries based on Lithium Ion chemistry. Each cell outputs 3.7V at 850mAh or 3.7V at 110mAh. The cells are terminated with a standard 2-pin JST-PH connector - 2mm spacing between pins. These batteries require special charging using a specialized LiPo charger. The battery includes built-in protection against over voltage, over current, and minimum voltage [29]. The battery is shown in Figure 27, it is small form-factor and lightweight.

![Lithium Ion Battery](image)

Figure 26: Lithium Ion Battery offering 3.7V at various mAh ranges (110, 850, etc.)

4.7 Technical Specifications of the Prototype Option Designs

What follows in Table 3 is a description about the technical specifications of each option presented in the “Component Options” section (components diagram). This guides us to choose the best components for our prototype based on our requirements, ensuring a successful prototype and streamlining development effort. We focus on low-power designs, consider form-factor, wireless support (BLE) and circuit-design features that will enable our collaborators to efficiently test and design prototypes as we advance through the product development cycle and aim to minimize budget and time-to-market.
### Comparing Technical Specifications of the Prototype Designs

<table>
<thead>
<tr>
<th></th>
<th>Option 1 (SparkFun Pro nRF52840 Mini)</th>
<th>Option 2 (Arduino Nano BLE)</th>
<th>Option 3 (Arduino Pro Mini + BT Module)</th>
<th>Option 4 (ATTiny85 uC + BT Module)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal Flash</td>
<td>1 MB</td>
<td>1 MB</td>
<td>32 kb</td>
<td>8 kB</td>
</tr>
<tr>
<td>Internal Ram</td>
<td>256 KB</td>
<td>256 KB</td>
<td>2 KB</td>
<td>512 B</td>
</tr>
<tr>
<td>Voltage (Component 1)</td>
<td>1.7V to 5.5V @ ~50 mA</td>
<td>3.3 V @ ~50 mA</td>
<td>3.3V</td>
<td>2.7V to 5.5V @ 40-100 mA</td>
</tr>
<tr>
<td>Voltage (Component 2)</td>
<td>N/A</td>
<td>N/A</td>
<td>3.3V @ 65 mA</td>
<td>3.3V @ 65 mA</td>
</tr>
<tr>
<td>Total Voltage</td>
<td>1.7V to 5.5V (3.3 is acceptable) @ ~50 mA</td>
<td>3.3 V @ ~50 mA</td>
<td>3.3V @ 200 mA (max)</td>
<td>3.3V @ 200 mA (max)</td>
</tr>
<tr>
<td>Dimensions (LxW)</td>
<td>2”x1”</td>
<td>1.7”x0.7”</td>
<td>0.7” x 1.3” (External BT will impact height)</td>
<td>1”x 0.5” (ATTiny uC will impact height)</td>
</tr>
<tr>
<td>Wireless Communication</td>
<td>Bluetooth 5, ANT, BLE</td>
<td>BLE</td>
<td><strong>External BT</strong></td>
<td><strong>External BT</strong></td>
</tr>
<tr>
<td>Peripherals</td>
<td>USB, UART, I2C, SPI</td>
<td>UART, I2C, SPI</td>
<td>UART, SPI, TWI</td>
<td>SPI, I2C</td>
</tr>
<tr>
<td>Online Support?</td>
<td>Great</td>
<td>Great</td>
<td>Good</td>
<td>Good</td>
</tr>
<tr>
<td>Other comments</td>
<td>Good form-factor</td>
<td>Good form-factor</td>
<td>Small</td>
<td>Extremely Small</td>
</tr>
<tr>
<td>Fast Prototyping?</td>
<td>Qwiic Connector makes for fast board setup, minimal soldering</td>
<td>Lacks Qwiic Connector - leads to slow prototype development (estimate <strong>1-2-month</strong> delay)</td>
<td>Advanced circuitry (soldering, wiring) will slow down development effort by <strong>1-2 months</strong></td>
<td>Advanced circuitry (layout, pcb, soldering, wiring) will slow down development effort by <strong>3-5 months</strong></td>
</tr>
<tr>
<td>Used by team member?</td>
<td>Yes, with great reviews</td>
<td>Yes, with good reviews</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Package Price</td>
<td>$29.95</td>
<td>$23.95</td>
<td>$9.95 + $25.95</td>
<td>$1.13 + $25.95</td>
</tr>
</tbody>
</table>

*Table 3: Comparison of Prototype Designs*

Based on the results of table 3, it is apparent that the nRF52840 is the best chip for our prototype. It is within budget and will allow us to implement our prototype quickly while...
maintaining the crucial aspects of our design, namely, ultra-low-power capabilities, low form-factor, technical support, scalability and adaptability to the skill-sets of the development team.

4.8 Implementation
This section will describe how the prototype was implemented. This assumes that Arduino is installed and operated on a Windows 10 desktop.

4.9 Set up Arduino Core
Go to “Preferences” and add the Adafruit package index JSON file to the URL of the Board Manager URL input (Appendix 1) to the Board Manager URL box. Go to Tools -> Board and select Board Manager. Search for nRF and install the “Adafruit nRF52” board package. To add a SparkFun Pro nRF52840 Mini to the boards list, a few adjustments need to be made to the Arduino core. The board definition needs to be set in order to help define pin-routing for the following serial buses and pins:

- Hardware Serial (Serial1) to pins 17 (TX) and 15 (RX) to match the Arduino Pro Mini pinout
- I²C to pins 8 (SDA) and 11 (SCL) required for the Qwicc connector
- SPI to pins 31 (MISO), 3 (MOSI), 30 (SCK)
- Built in LED (LED_BUILTIN) to pin 7

The files are downloaded and stored in the nRF52 core installation and stored in the “nrf52” version sub-directory (i.e. 0.20.5). The “boards.txt” file needs to be edited and the section in Appendix 2 is appended to the bottom to define pin-routing. Finally, the contents of the “variant” folder from the folder that was downloaded from the CDN into the original directory and sample code (Appendix 3) can be run to test the device.

4.10 Quantized Data Transfer
The data points we are examining are heart rate and oxygen saturation. We will first examine the metadata to create a byte level representation of the data. There are 8 bits in a byte and there are 4 bits in a nibble. A hexadecimal byte will contain two characters (i.e. 0xFF is equivalent to ‘1111 1111’). The range for a hexadecimal byte will be 0 to 255 (holding 256 values in total, including 0).

---

1 These files can be found at the following link: https://cdn.sparkfun.com/assets/learn_tutorials/8/4/2/sparkfun-nrf52840-arduino-board-def-v1.4.zip
<table>
<thead>
<tr>
<th>Data Point</th>
<th>Unit</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oxygen Saturation</td>
<td>Percentage (%)</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>Heart Rate</td>
<td>Beats Per Minute (BPM)</td>
<td>0</td>
<td>200</td>
</tr>
<tr>
<td>Counter</td>
<td>N/A</td>
<td>1</td>
<td>255</td>
</tr>
</tbody>
</table>

Table 4: Thresholds for measurements to be quantized.

4.10.1 Oxygen Measurement converted to binary example:
Examining a few potential oxygen measurements, we prove that we require a minimum of 7 bits to display our oxygen measurement.

- 50: 0b0011 0010
- 52: 0b0011 0100
- 53: 0b0011 0101
- 99: 0b0110 0011
- 100: 0110 0100

Next, look at our Heart Rate measurements to explore the minimum number of bits required.

- 27: 0b0001 1011
- 200: 0b11001000

Heart rate requires a minimum of 8 bits to represent. From 0 to 200. In total, to represent both oxygen and heart rate, 15 bits (7+8) are required. This can be represented as 4 hex characters. Let us look at an example. If the oxygen measurement is 99 and the heart rate is 77, this would be represented in binary as follows:

Oxygen -> 99 (decimal) -> 1100011
Heart Rate -> 77 (decimal) -> 0100 1101

And combining these together by concatenation (oxygen concatenated to heart rate), we get the 15-bit binary masked representation with an added least significant bit:

Oxygen + Heart Rate: 1100 0110 1001 1010

The 1\textsuperscript{st} (from least significant bit) to the 9\textsuperscript{th} bit are the heart rate measurement. The 10\textsuperscript{th} to the 16\textsuperscript{th} bit (most significant bit) are the oxygen measurements. Representing this in hex would result in:

Hex representation -> 0xC60D

This can be sent via Bluetooth serial in just 4 characters. The structure and schema is maintained and all the data from a single measurement is grouped together. Let us look at another sample measurement:
Oxygen 100, Heart Rate 70, which turns into:
Oxygen -> 100 -> 110 0100
Heart Rate -> 70 -> 0 0100 0110
Combined -> 1100 1000 0100 0110
Hex representation -> 0xC846

This method of storing and sending data minimizes the amount of storage needed to hold and send data. It is a common technique to expand and contract data in cutting-edge IoT applications.

4.10.2 How much data can we store in the RAM?
Our device has 256 kb of ram. This translates to roughly 262,144 bytes (note: we will equate a kilobyte to 1024 bytes to maintain accuracy although commonly a ‘kilo’ is 1000. This is because a computer operates in base 2). For example, our prototype can record once every 1 minute, resulting in 1440 (24 hours X 60 minutes) measurements occurring in 24 hours. Each recorded measurement will be stored in 2 bytes (without timestamp counter or checksum) resulting in 2880 bytes (1440 measurements * 2 bytes) being stored daily.

Recall that the size of our RAM is 256 kB. This means that we can store approximately 91 days of data (262,144 bytes divided by 1day/2880 bytes) in our device before we reach our capacity, assuming that we are storing 2 bytes. To explore different variations, Table 5 shows the different amounts of data we can store based on varying recording interval and data packet sizes (in case of adding counter and checksum). Note that we have access to the exact amount of ram available for our board – 248,832 bytes. We will use this number for our calculations – not 256,000 bytes or 262,144 bytes.
### How many Days before a burst transmission is needed?

<table>
<thead>
<tr>
<th>Recording Interval</th>
<th>Data Packet Size (Bytes)</th>
<th>Measurements per Day</th>
<th>Bytes per Day (Size * # of measurements per day)</th>
<th>How many days before a data transfer is needed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 second</td>
<td>2</td>
<td>86400</td>
<td>172800</td>
<td>14.4</td>
</tr>
<tr>
<td>10 seconds</td>
<td>2</td>
<td>8640</td>
<td>17280</td>
<td>144.0</td>
</tr>
<tr>
<td>1 minute</td>
<td>2</td>
<td>1440</td>
<td>2880</td>
<td>864.0</td>
</tr>
<tr>
<td>10 minutes</td>
<td>2</td>
<td>144</td>
<td>288</td>
<td>8640.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 second</td>
<td>4</td>
<td>86400</td>
<td>345600</td>
<td>7.2</td>
</tr>
<tr>
<td>10 seconds</td>
<td>4</td>
<td>8640</td>
<td>34560</td>
<td>72.0</td>
</tr>
<tr>
<td>1 minute</td>
<td>4</td>
<td>1440</td>
<td>5760</td>
<td>432.0</td>
</tr>
<tr>
<td>10 minutes</td>
<td>4</td>
<td>144</td>
<td>576</td>
<td>4320.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 second</td>
<td>6</td>
<td>86400</td>
<td>518400</td>
<td>4.8</td>
</tr>
<tr>
<td>10 seconds</td>
<td>6</td>
<td>8640</td>
<td>51840</td>
<td>48.0</td>
</tr>
<tr>
<td>1 minute</td>
<td>6</td>
<td>1440</td>
<td>8640</td>
<td>288.0</td>
</tr>
<tr>
<td>10 minutes</td>
<td>6</td>
<td>144</td>
<td>864</td>
<td>2880.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 second</td>
<td>8</td>
<td>86400</td>
<td>691200</td>
<td>3.6</td>
</tr>
<tr>
<td>10 seconds</td>
<td>8</td>
<td>8640</td>
<td>69120</td>
<td>36.0</td>
</tr>
<tr>
<td>1 minute</td>
<td>8</td>
<td>1440</td>
<td>11520</td>
<td>216.0</td>
</tr>
<tr>
<td>10 minutes</td>
<td>8</td>
<td>144</td>
<td>1152</td>
<td>2160.0</td>
</tr>
</tbody>
</table>

*Table 5: Trade-off - battery vs. memory storage (recording intervals).*

Table 5 proves that the limiting factor of our prototype’s data storage capabilities will be the battery in most cases. In the special circumstance where a patient needs to record every minute at a data packet size of 6 bytes, it will still successfully hold the data points before the battery runs out.

#### 4.10.3 Timestamp

We will include a counter that will record measurements every minute. We can expect the counter to equal the number of measurements and the maximum reasonable number of measurements will be from 144 to 1440. This will require at least 11 bits (0 – 2047).

#### 4.10.4 Checksum:

It is possible to incorporate a checksum or check even bit to ensure the integrity of data. In our case, we can add a last bit that will be enabled or disabled to ensure an even number of 1’s. We will incorporate this functionality in the future.
4.10.5 Implementation:
The implementation of this requires an understanding of computer operating systems. In a computer operating system, at its core, data is stored in bits (0’s and 1’s). What we interact with is layers upon layers of abstraction from that core functionality.

For our purpose, we will do a few interesting things. Firstly, we will use the information we have gathered about the ranges of our data to set the structure of our final measurement. Recall that we were able to achieve all possible values of our data in a four-byte hexadecimal representation – the first 7 bits being the oxygen measurement and the last 9 bits representing the heart rate measurement.

From our sensor, we receive data in decimal format. To achieve the desired representation, we will shift the oxygen measurement 9 bits to the left. This process is shown below:

1. We record an Oxygen Saturation measurement of 99 and heart rate of 77.
2. We translate the Oxygen Saturation to its binary representation of ‘110 0011’
3. Shift to the left 9 bits to make room for our heart rate measurement. This results in ‘1100 0110 0000 0000’
4. Convert our heart rate measurement to binary: 100 1101
5. Perform the logical ‘OR’ operation on the result of Step 3, using the converted heart rate measurement value.
6. The resulting binary number will be: 1100 0110 0100 1101
7. 1100011001001101
8. Converting that to hexadecimal will give us: 0xC64D
9. Converting that to decimal will result in a number that holds our measurements: 50765

Note, in our software, we do not need to convert our numbers to binary. The operating system already holds our data in binary. This means that only two operations are required. A leftward shift of 9 bits, and a logical ‘OR’ operation. The pseudocode follows:

```plaintext
final = 0;
ox_sat = 99;
heart_rate = 77;
placeholder = ox_sat << 9; //shift bits to the left to make room for heart rate
final = placeholder | heart_rate; //perform an ‘or’ to combine heart rate
print(final); // a large decimal number. The reverse operations will unravel it
```

4.10.6 Range of Oximeter:

normal human being: 97 and above with issues: <95 below; idiopathic pulmonary fibrosis.
Device will be on all the time, but when you receive an instruction from the app (when the app opens), the device must send data. Until then it is in sleep mode. Every minute it is sleeping but will wake up and measure.
4.11 Optimizing Code

The code is presented in Appendix 5. The sketch uses 9% of program storage space. This is 77,520 bytes out of a maximum 815104 bytes. We will examine ways to minimize the program storage space. First, we examine our main code. Our main code is less than 200 lines long and uses a classical “while” loop approach like most embedded systems. We remove serial print lines, examine primitives to minimize usage and then look into timing mechanisms.

4.11.1 Remove Serial Print lines

Removing serial print lines causes the program size to decrease by about 528 bytes. This is a minimal change, but it is important to optimize code to occupy the minimum amount of program storage space.

4.11.2 Examine the “types” being used and minimize to smaller primitives

Next, we will look at the primitive types being used in our code. The goal of this effort is to allocate less space in memory used by variables if needed. Usually, numbers are assigned “int” values for no good reason. An int primitive can hold 2 bytes of data (16 bits long). If we know that a variable will be, for example, appearing in the range of 1-20, then a shorter type can be used to save program memory. We implemented this approach throughout our code and found no significant improvements in program storage space.

4.11.3 Remove smart-timers and use smart delays

When we enter sleep or idle and activate the sensor, we are utilizing a smart-timer at the end of our loop. This smart-timer causes the system to consume CPU and raises the current consumption. A smart delay reads the milliseconds at a certain time and avoids (turning on the sensor, in our case) until a time interval has been exceeded. Technically, this involves reading the time in milliseconds, setting an interval (i.e. 1000 ms), and continuously subtracting the current time from the original time until the interval has been exceeded.

We use the term exceeded because an operation might cause the system to become busy, in which case our millisecond timing might exceed its interval limit. For example, if our interval is 1000 ms, our system might be busy when we reach our check. In this case, even if we read a value of 1200 ms, we will still be safe. The only benefit is that it allows other operations to continue. Instead of using the smart-timer, we will utilize the delay function. The overarching benefit of this type of delay is not in its precision. Instead, it is that it prevents stalls in the program that a delay does cause.

However, the nRF52840 has an automated peripheral power management functionality that is triggered by the delay function. When a delay is called, all inactive peripherals are shut down to achieve ultra-low power with minimal supervision. This is the delay we will use instead of the smart-timer in order to fulfill the power requirements of our design.
4.12 Power Profiling
Power meters are used to create a power profile of an embedded system measuring current flow as well as voltage and other measurements. The power meter will be used in our design to minimize the current consumption such that we achieve an ultra-low power design.

4.12.1 USB Power Meter
Options for power meters are shown in Table 6. All are within the required budget. We assess the precision of the device to record the microamp (μA) range.

<table>
<thead>
<tr>
<th>Product</th>
<th>Requirements: Does it detect μA or mA?</th>
<th>Record Data Option?</th>
<th>Precision</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spark Fun [34]</td>
<td>No</td>
<td>Yes</td>
<td>3</td>
<td>24.95</td>
</tr>
<tr>
<td>MakerHawk [35]</td>
<td>No</td>
<td>Yes</td>
<td>3</td>
<td>11.59</td>
</tr>
<tr>
<td>MakerHawk [36]</td>
<td>Yes</td>
<td>No</td>
<td>4</td>
<td>13.99</td>
</tr>
<tr>
<td>KWS-MX18 [37]</td>
<td>No</td>
<td>No</td>
<td>3</td>
<td>13.60</td>
</tr>
</tbody>
</table>

Table 6: Power Meters for power profiling.

The MakerHawk with 4 decimal place precision is most suited for our design and will be used to create a power profile of our system.

4.13 Profile (Current Consumption)
Table 7 shows the power profile of the board. We conclude with a “resting” state of the board at 2.3 mA (95.6 hours of battery life using 3.7V Li-On battery at 220 mAh), a brief “Sensor Active” state at 16 mA and a “Sensor Reading” state at 22.5 mA. The active and reading states last for approximately 10 seconds. For example, if the device is set to read every 10 minutes, the consumption will be 2.3 mA for 9 minutes and 50 seconds. When the device turns on the sensor and reads, the consumption will range from 16-22.6 mA in a 10-15 second measurement window.
<table>
<thead>
<tr>
<th>Step taken</th>
<th>mA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery Mode without Sensor (w/ Smart Timer)</td>
<td>19</td>
</tr>
<tr>
<td>Discovery Mode with Sensor Activated (w/Smart Timer)</td>
<td>25</td>
</tr>
<tr>
<td>Paired Mode without Sensor (w/Smart Timer)</td>
<td>19.2</td>
</tr>
<tr>
<td>Paired Mode with Sensor Activated (w/Smart Timer)</td>
<td>24.7–25</td>
</tr>
<tr>
<td>Using delays instead of smart timers</td>
<td>13.9</td>
</tr>
<tr>
<td>Without Bluetooth LED (Blue)</td>
<td>13.7</td>
</tr>
<tr>
<td>Without Bluetooth Code</td>
<td>13.2</td>
</tr>
<tr>
<td>With Bluetooth Code</td>
<td>13.7</td>
</tr>
<tr>
<td><strong>Resting State - after Removing Power LED and connecting sensor using Transistor (resting state)</strong></td>
<td><strong>2.3</strong></td>
</tr>
<tr>
<td><strong>Sensor Active (2-4 seconds)</strong></td>
<td><strong>16</strong></td>
</tr>
<tr>
<td><strong>Sensor Performing a &quot;read&quot; (4-6 seconds)</strong></td>
<td><strong>22.5</strong></td>
</tr>
</tbody>
</table>

*Table 7: Steps taken to characterize the power profile of the nRF52840.*

These results are convincing and can be improved upon in the future. The voltage regulator and other board layout elements can be removed along with any unnecessary components that a chip designer may do without.

5. Results
The final results of our prototype are a design with all components fitting inside a 2”x1” inch length and width (with ample room for PCB layout improvements), a steady current consumption of 2.3 mA achieving up to 95 hours of battery life, Bluetooth connection and configurable settings for future FDA approval efforts.

6. Discussion
Throughout our development we made changes to various aspects of our design to achieve the desired functionality. One major change was switching from smart timers to delays. We found that using “smart” delays on modern microcontrollers have significant drawbacks. Smart delays poll the milliseconds function until enough time has elapsed. This is a simple loop that reads the time in milliseconds to see if the required interval has been reached and enables other functions to go on – like sensor readings and mathematical calculations. In some cases, chip designers cause the microcontroller to go into sleep mode during a delay call. No other reading of sensors, mathematical calculations or pin manipulation can go on during the delay function, so in effect, it brings most other activities to a halt.

However, in modern microcontrollers, smart power management is embedded into the core of the device. If a delay is called, the device enters “sleep” mode, and all unnecessary functions are halted. In our design, we delay for 5 seconds repeatedly until we determine that a Bluetooth command has been sent or if it is time to perform a new reading.
Furthermore, we contributed to the library by using the reference guides to determine how to shut down the sensor’s MAX30101 sensor. This was done with the following command.

```cpp
//This command shuts down the MAX30101 sensor.
bioHub.writeRegisterMAX30101(0x09, 0x80);
```

We also wrote a function to shut down the MAX32664 (the photodiode sensor) using the following block of code:

```cpp
uint8_t SparkFun_Bio_Sensor_Hub::shutdownMAX32664(){
  uint8_t statusByte = writeByte(SET_DEVICE_MODE, 0x00, SHUTDOWN);
  if (statusByte != SUCCESS )
    return statusByte;
  // Here we check the status of the controller
  uint8_t responseByte = readByte(READ_DEVICE_MODE, 0x00); // 0x00 only possible Index Byte
  return responseByte; // This is the status byte
}
```

Lastly, we utilized the nrF header file found in Appendix 4 to use system-level commands to force sleep and set low power modes.

```cpp
// NRF Commands
  __SEV(); //Sleep
  __WFE(); // Wait for event
  NRF_POWER->TASKS_LOWPWR = 1; // Set device to ultra-low-power mode
  NRF_POWER->SYSTEMOFF = 1; //Turn device off
```

However, since the developers at SparkFun designed their own version of this board, some of these functionalities were not implemented at the abstraction layer we desire. These functionalities were also conclusive – if the board entered deep sleep mode (at 1uA) then an external trigger needed to cause an awake. Likewise, if the board entered system off mode, then an external trigger would be required to reset the system and that would raise the cost of our product and introduce additional circuitry.

7. Conclusion
We tested the resulting system using Bluetooth applications on a mobile device and on a portable computer to verify that the data was sent accurately. We designed a prototype of a lightweight (small form-factor) pulse oximeter on a small circuit board operating without a display and other current inducing elements at low-power. Our prototype sends data wireless
and features wireless flexibility and opportunity for wireless charging (through the lithium ion batteries currently in use).

8. Future Work
Development effort is required on a few fronts. Firstly, the system must be designed to efficiently burst transmit data and accommodate Bluetooth communications with a mobile device while maintaining an even lower power-profile.

An analytics pipeline using data engineering principles should be implemented to uncover insights that might relate demographic (ethnicity and cultural background, age, gender) and individualized patient data (i.e. height, weight, blood pressure) in a meaningful way to more closely define the parameters that characterize certain diseases and blood oxygen complications.

FDA approval is necessary for reputation and a strong investor presence as the product approaches the market. Algorithms on-board the MAX32664 must be fine-tuned, calibrated, and adapted to different skin tones and environments. The FDA approval process requires testing among populations with various pigmentation and a clinical trial should be designed to ensure that the device runs accurately and reliably.

PCB design will aid in decreasing the form-factor even more while improving the layout of the device such that it can be placed on a flexible substrate or more convenient medium for patient comfort and consistent wearing.

Feedback loops connecting this device to oxygen tanks should be explored in an effort to autonomously allow this system to monitor patients’ oxygen levels and adjust oxygen tank gauges without human intervention. This would prevent deaths caused by hypoxia and allow pattern recognition algorithms to determine the “oxygen routine” of patients.
9. References:


10. Appendix

[1] https://www.adafruit.com/package_adafruit_index.json

[2] Setting the pin layout of the nRF52840 board

/*#***************************************************************************/
/*# SparkFun Pro nRF52840 Mini
#***************************************************************************/
sparkfunnrf52840mini.name=SparkFun Pro nRF52840 Mini

# DFU Mode with CDC only
sparkfunnrf52840mini.vid.0=0x1B4F
sparkfunnrf52840mini.pid.0=0x002A

# DFU Mode with CDC + MSC (UF2)
sparkfunnrf52840mini.vid.1=0x1B4F
sparkfunnrf52840mini.pid.1=0x0029

# Application with CDC + MSC
sparkfunnrf52840mini.vid.2=0x1B4F
sparkfunnrf52840mini.pid.2=0x8029

# CircuitPython
sparkfunnrf52840mini.vid.2=0x1B4F
sparkfunnrf52840mini.pid.2=0x802A

sparkfunnrf52840mini.bootloader.tool=bootburn

# Upload
sparkfunnrf52840mini.upload.tool=nrfutil
sparkfunnrf52840mini.upload.protocol=nrfutil
sparkfunnrf52840mini.upload.use_1200bps_touch=true
sparkfunnrf52840mini.upload.wait_for_upload_port=true
#sparkfunnrf52840mini.upload.native_usb=true

# Build
sparkfunnrf52840mini.build.mcu=cortex-m4
sparkfunnrf52840mini.build.f_cpu=64000000
sparkfunnrf52840mini.build.board=NRF52840_FEATHER
sparkfunnrf52840mini.build.core=nRF5
sparkfunnrf52840mini.build.variant=sparkfun_nrf52840_mini
sparkfunnrf52840mini.build.extra_flags=-DNRF52840_XXAA {build.flags.usb}
sparkfunnrf52840mini.build.vid=0x1B4F
sparkfunnrf52840mini.build.pid=0x5284
# SoFDevice Menu
# Ram & ROM size varies depending on SoFDevice (check linker script)

sparkfunnrf52840mini.menu.softdevice.s140v6=s140 6.1.1 r0
sparkfunnrf52840mini.menu.softdevice.s140v6.build.sd_flags=-DS140
sparkfunnrf52840mini.menu.softdevice.s140v6.build.sd_name=s140
sparkfunnrf52840mini.menu.softdevice.s140v6.build.sd_version=6.1.1
sparkfunnrf52840mini.menu.softdevice.s140v6.build.sd_fwid=0x00B6
sparkfunnrf52840mini.menu.softdevice.s140v6.build.ldscript=nrf52840_s140_v6.ld
sparkfunnrf52840mini.menu.softdevice.s140v6.upload.maximum_size=815104
sparkfunnrf52840mini.menu.softdevice.s140v6.upload.maximum_data_size=248832

# Debug Menu
sparkfunnrf52840mini.menu.debug.l0=Level 0 (Release)
sparkfunnrf52840mini.menu.debug.l0.build.debug_flags=-DCFG_DEBUG=0 -Os
sparkfunnrf52840mini.menu.debug.l1=Level 1 (Error Message)
sparkfunnrf52840mini.menu.debug.l1.build.debug_flags=-DCFG_DEBUG=1 -Os
sparkfunnrf52840mini.menu.debug.l2=Level 2 (Full Debug)
sparkfunnrf52840mini.menu.debug.l2.build.debug_flags=-DCFG_DEBUG=2 -Os
sparkfunnrf52840mini.menu.debug.l3=Level 3 (Segger SystemView)
sparkfunnrf52840mini.menu.debug.l3.build.debug_flags=-DCFG_DEBUG=3 -Os

[3] Sample code to run the SparkFun Pulse Oximeter

/*
This sketch reads heart rate and blood oxygen levels.

SDA -> SDA
SCL -> SCL
RESET -> PIN 4
MFIO -> PIN 5

If you run into an error code check the following table to help diagnose your problem:
1 = Unavailable Command
2 = Unavailable Function
3 = Data Format Error
4 = Input Value Error
5 = Try Again
255 = Error Unknown
*/

#include <Wire.h>
#include <bluefruit.h>
#include <PulseOxy_Sensor.h>

// Reset pin, MFIO pin
const uint8_t resPin = 4;
const uint8_t mfioPin = 5;

// Takes address, reset pin, and MFIO pin.
SparkFun_Bio_Sensor_Hub bioHub(resPin, mfioPin);
BLEUart bleuart; // uart over ble
bioData body;

// BioData is a "struct".
// body.heartrate - Heartrate
// body.confidence - Confidence in the heartrate value
// body.oxygen - Blood oxygen level
// body.status - Has a finger been sensed?

void setup() {
    // Initialize hardware:
    Serial.begin(115200);

    // Uncomment the code below to disable sharing
    // the connection LED on pin 7.
    // Bluefruit.autoConnLed(false);

    // Initialize Bluetooth:
    Bluefruit.begin();
    // Set max power. Accepted values are: -40, -30, -20, -16, -12, -8, -4, 0, 4
    Bluefruit.setTxPower(4);
    Bluefruit.setName("Pulse-HR-Sensor");
    bleuart.begin();

    // Start advertising device and bleuart services
    Bluefruit.Advertising.addFlags(BLE_GAP_ADV_FLAGS_LE_ONLY_GENERAL_DISC_MODE);
    Bluefruit.Advertising.addTxPower();
    Bluefruit.Advertising.addService(bleuart);
    Bluefruit.ScanResponse.addName();
    Bluefruit.Advertising.restartOnDisconnected(true);
// Set advertising interval (in unit of 0.625ms):
Bluefruit.Advertising.setInterval(32, 244);
// number of seconds in fast mode:
Bluefruit.Advertising.setFastTimeout(30);
Bluefruit.Advertising.start(0);

Wire.begin();
boolean result = bioHub.begin();
if (result != 0) // Zero errors!
  Serial.println("comm error");

boolean error = bioHub.configBpm(MODE_ONE); // Configuring the BPM settings.
if (error != 0){ // Zero errors!
  Serial.print("Err");
  Serial.println(error);
}

// Data lags a bit behind the sensor, if your finger is on the sensor when
// it's being configured this delay will give some time for the data to catch
// up.
//Serial.println("Loading up the buffer with data....");
delay(4000);

}

const long interval = 1000;
unsigned long previousMillis = 0;

void loop() {

  unsigned long currentMillis = millis();

  if(currentMillis - previousMillis >= interval){
    previousMillis = currentMillis;
    if (bleuart.available()) {
      uint8_t char_input;
      char_input = (uint8_t) bleuart.read();

      switch (char_input) {
        case 0: //Trigger burst transmission
        case 'burst':
          break;
      }
  }
case 1: //Configure recording Interval
    case '10m': //10 minutes
        break;
    case '1m': //1 minute
        break;
    default:
        break;
}

body = bioHub.readBpm();
if(body.oxygen >= 0 && body.oxygen <= 100 && body.heartRate >= 0 && body.heartRate <= 200){
    Serial.print("HR: ");
    Serial.println(body.heartRate);
    Serial.print("Ox: ");
    Serial.println(body.oxygen);
}

// Send heart rate and oxygen on bluetooth
// Data compression and expansion design allows data transfer in single packets
// Does not include a counter. Will be included later.
uint8_t oxSat = body.oxygen;
uint8_t heartRate = body.heartRate;

uint8_t packet[4] = {};
packet[0] = oxSat;
packet[1] = heartRate;
packet[2] = 0; //Counter
packet[3] = 0; //Counter

//for Debug
//Serial.println(packet[0], HEX);
//Serial.println(packet[1], HEX);

//Send it off via Bluetooth.
bleuart.write(packet, sizeof(packet));
}

    /*************************************************************************/
__IOM uint32_t INTENCLR;  /*! (@ 0x0000030) Disable interrupt */
__IM uint32_t RESERVED4[61];
__IOM uint32_t RESETREAS;  /*! (@ 0x00000400) Reset reason */
__IM uint32_t RESERVED5[9];
__IM uint32_t RAMSTATUS;  /*! (@ 0x00000438) Deprecated register - RAM status register */
__IM uint32_t RESERVED6[3];
__IM uint32_t USBREGSTATUS;  /*! (@ 0x00000500) System OFF register */
__IM uint32_t RESERVED7[49];
__IOM uint32_t SYSTEMOFF;  /*! (@ 0x00000510) Power-fail comparator configuration */
__IM uint32_t RESERVED8[3];
__IOM uint32_t POFCON;  /*! (@ 0x0000051C) General purpose retention register */
__IM uint32_t RESERVED9[2];
__IOM uint32_t GPREGRET;  /*! (@ 0x00000520) General purpose retention register */
__IOM uint32_t GPREGRET2;  /*! (@ 0x00000578) Enable DC/DC converter for REG1 stage */
__IM uint32_t RESERVED10[21];
__IOM uint32_t DCDCEN;  /*! (@ 0x00000580) Enable DC/DC converter for REG0 stage */
__IM uint32_t RESERVED11;
__IOM uint32_t DCDCENO;  /*! (@ 0x00000640) Main supply status */
__IM uint32_t RESERVED12[47];
__IM uint32_t MAINREGSTATUS;  /*! (@ 0x00000900) Unspecified */
__IM uint32_t RESERVED13[175];
__IOM POWER_RAM_Type RAM[9];  /*! Size = 2448 (0x990) */
*/
} NRF_POWER_Type;  /*! Size = 2448 (0x990) */