Servo and Sensor Enhanced Smart Arm Cast

A Major Qualifying Project submitted to the faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
degree of Bachelor of Science

Submitted by:
Aedan Fahy [af56@comcast.net]
Achilles Gikas [achillesgikas@outlook.com]
Shaniqua Miller[shaniquamiller55@gmail.com]
Shawn Mills [shawnjr@shawnpmillsmdpc.com]

25 April 2024

Project Advisors:
Mehul Bhatia

Project Co-Advisors:
Songbai Ji

Table of Contents

List of Tables & Figures

Abstract

Modern arm casting techniques require skin-irritating and non-reusable material with tedious multi-step processes to treat patients with arm fractures or other ailments. This project aims to create a new smart arm casting device, requiring a single mold capable of tightening autonomously while also providing biometric data for patients. Using Thermoplastic Polyurethane (TPU), the cast is flexible and durable enough to be protective, tightened and loosened, and light in weight. The adjustment of tension is performed by a series of intertwining fibers attached to a servo motor, which is supported by pressure and heart rate sensors attached to an Arduino controller. From this, the cast is capable of providing consistent tightness to the patient's arm, while also obtaining biometric data helpful to patients and caretakers. Furthermore, the potential use of this product could help eliminate extra steps in the current casting process and innovate data and devices in the healthcare industry.

1.0 Introduction

Being one of the most complex bone structures in the human body, the wrist and lower hand area of the body consists of a multitude of bones that grow and change over the course of a person's early childhood. These bones however are incredibly susceptible to injuries, making up an average of 2.6 million Emergency Room visits in the US every year in the past decade. As of 2018, almost 793,000 patients suffering a hand or wrist injury were ages 0 to 19 years, which raises concern considering bone structure changes through age (Gordon, 2021). While the severity of these injuries varies, nearly all wrist fractures require the use of a cast to immobilize and restrict the wrist from moving. After initial covering and casting, the patient is left to work with their primary care options, with recovery in the area of 6-12 weeks depending on the severity of the injury (O'Hara et al., 2021).

Immobilization casting of injured limbs has been performed for thousands of years. Before modern-day casting materials became widely used, many doctors used a variety of materials to form rigid casts. Currently, the most commonly used material for casting of all types is Plaster of Paris. This material is produced by removing the impurities from the main gypsum and then heating it under controlled conditions to reduce the amount of water during crystallization (Simmons & Cox, 1957). However, this form of casting comes with a variety of issues, from minor skin irritation to compartment syndrome (a medical condition characterized by increased pressure within a muscle compartment), joint stiffness, and pressure sores.

With the rapid advancement of technology today, it is critical to think about how specific products can be changed to make life more efficient and better quality. One of the industries that could benefit from the use of a technological smart device is healthcare. The current casting processes either use time-consuming methods such as custom molding or implementation of complex tightening mechanisms that are difficult for both medical staff and patients to use, resulting in multiple visits to the hospital and increased time of recovery. These processes can be made far more efficient by replacing the complex multi-strap and manual tightening mechanisms as well as the time consuming custom molding by creating a motor-driven system that can tighten the casting material around a person's arm on its own with the use of smart technology. To create a feasible design, a number of factors must be considered, including what specific mechanical equipment and material is required, how the assembly will operate correctly and efficiently, how expensive it is to create such a device, and how safe and comfortable it is to use.

A smart device is considered to be a complex system consisting of miniature mechanical and electrical parts such as sensors, data storage, microprocessors, software, and connectivity that are combined with certain hardware to operate autonomously. The utilization of smart devices in society has been constantly improving and expanding due to the growth of information technology and processing power combined with science that decreases the size of various devices (Porter & Heppelmann, 2014). Innovation in smart technologies has led to a wide scale of technological consumer products such as smart cars, smartphones, smart speakers, smart laptops and tablets, and smart health monitoring devices. Not only do smart devices make human lives easier through their convenience factor, but they also utilize big data to create fast, accurate, and personalized information for their users. According to an article from the journal *mHealth*, "Smart devices that analyze data to provide personal exercise and health conditions, checkups, and information, are making our lives easier … as it grows into an expected industry in the future" (Son & Kwon, 2024). This further supports the idea of creating a smart device that innovates the casting process in the healthcare industry, it also supports the use of sensor technology that exhibits biometric data to personally manage an injured human's health.

To better understand the technique of creating a smart arm cast device, the product was initially discussed and designed to be put together in three layered sleeves that will wrap around a human arm. The innermost layer will incorporate the use of two flat pressure sensors and a cotton sock that will touch directly to the skin of a person's injured limb. Doing this will allow users to accurately gauge a person's pressure data to communicate how much the cast should compress with an Arduino controller while also providing a sense of comfort and cushion.. Around this initial layer, a second sleeve made from a more protective biocompatible material will hold more mechanical equipment including an Arduino Nano R3 board that connects to both of the pressure sensors, a servo motor, an external pulse oximeter, a Liquid Crystal Display (LCD), and the battery that powers the system. This layer will also prevent potential injuries on the skin from mechanical equipment while also providing slots for each of the equipment to hold in place. The Arduino board will have its own source code that will allow the pressure sensors to work alongside the motor so that the motor will understand when it needs to stop rotating, thus incorporating the use of Arduino coding to control the limit of how much pressure there is when the cast tightens around a person's arm. Additionally, code is programmed to control the pulse oximeter by correctly measuring the blood oxygen level and heart rates of the patient while being displayed visually via the LCD system. Finally, the outermost layer would have a pulley wheel that spins based on the power of the motor and will rotate a nylon fish wire string that intertwines throughout the holes of the cast similar to a shoelace, so that it tightens effectively. An external button that connects to the Arduino will be programmed for the user to be able to recalibrate the overall pressure of the system patient. With the design in mind and drawn out, the computer-aided design software, SolidWorks, helped visualize a 3-D model of what the smart device will look like with all components considered.

Various design iterations were constantly made throughout the process to create our final smart cast hardware model. One of the most significant design changes was the incorporation of holes and an extended wall with holes extruding from the second layer that would be used to tighten the cast with nylon fish wire similar to a shoelace weaving through respective eyelets. In addition, a bottom and shelled-out loft was needed to hold the large battery and Arduino together so it could properly sit beneath a person's arm without getting in the way, being protected by its casing, and not being visible. The incorporation of these features made it unable to create an overall third protective layer that sits around the new second outer layer. The new design concept for the smart cast device would have just the inner layer of cotton cushion and the newly improved second layer with casing to protect external mechanical equipment such as the servo motor, LCD, and emergency stop button, all including their respective wiring.

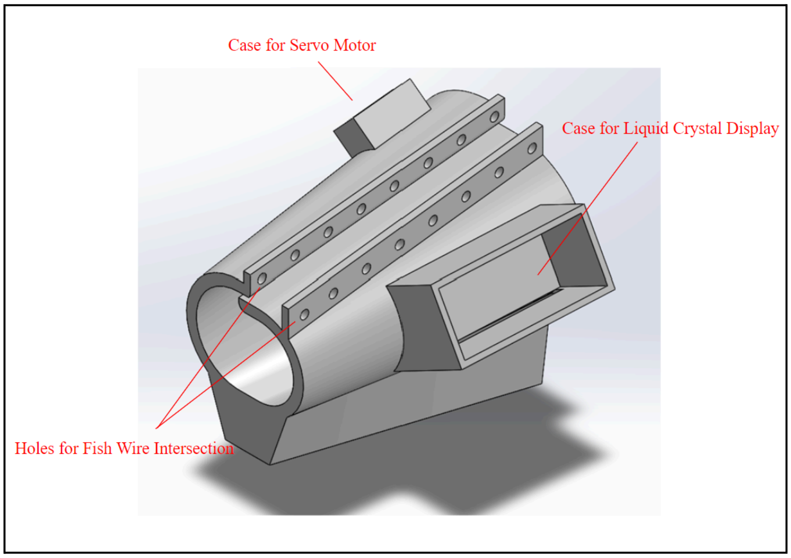


Figure 1: Final Cast Body Design Iteration

A major goal that this project can lead towards is further endeavors into sensor-based techniques in casting. Along with this, we also aim to introduce this form of implementing smart wearable casting into the healthcare industry. While the objectives of this project focus on developing an improved arm cast capable of sending biometric data directly to the patient, these objectives can be readily accomplished in other forms of limb immobilization. The most immediate continuation of this project could be in the form of sensor-based casting for other common fracture injuries such as in the foot or ankle. These improvements aim to better the track and timeline of patient recovery by providing improved biometric readings to the patient as well as changing the current casting process to make it less difficult and more reusable.

2.0 Background

Based on research and the history of casting, this section is written to provide context to the pre-existing subjects and components of the smart cast project as well as an introduction to the healthcare applications, methodologies, and materials used within the industry.

2.1 Conventional Arm Casting

The use of casting material, plaster of Paris, has been used for centuries, typically for wall plastering in buildings. This use of the material dates back to the times of ancient civilizations such as the Egyptians and early Romans. In the early eighteenth century, this material began being used to treat injured limbs by fixing the appendage in a heavy wooden case filled with plaster (Szostakowski et al., 2017). Afterwards, further developments were made for treating injured people including applying plaster material to linen strips to form plaster bandages. During the 1850s, a Dutch military surgeon, Anthonius Mathijsen, found that plaster bandages soaked in water can create a hard solid cast that can be used for injured limbs. This method of casting began being used for nearly a hundred years afterward (Szostakowski et al., 2017).

The purpose of the arm cast is to fix the arm in a comfortable immobilized fashion so that the bone can heal itself properly. In the United States alone, approximately 3.5 million civilians have to visit a hospital for an arm-related fracture every year (The Ohio State University, 2016). Currently, there is a critical need for a smart wearable arm cast to facilitate better patients' and doctors' needs as well as make the casting process more efficient. This is evident according to many hospital's opinions including the Penn Medicine Chester County Hospital which writes in its blog, "Having a cast is never fun - but the attention you receive with it can be … physically, you may experience a lot of discomfort and pain" (Chester County Hospital, 2019). It is a common complaint among patients who feel that casts are uncomfortable and still do not get rid of the pain completely, even though they are currently the most efficient way of healing fractured bones. In addition to this issue, the current casting process is also difficult to deal with for both medical staff and the injured person. The most common way casts are applied to a patient is through wrapping the injured limb in both a sock and cotton wrap, then wrapping the area with material such as plaster or fiberglass, and then officially molding the cast to achieve immobility (The Center Oregon). Although this process seems tedious, the time spent to mold a cast can be greatly reduced and potentially eliminated.

2.2 Existing Casting Techniques

Some companies' products such as the BOA® Fit System have a mechanism with the use of fibers that can be tightened similar to shoelaces in order to achieve a snug and comfortable fit around the arm. This device aids the project in the design process by thinking about how to compress a sleeve around a person's lower arm. With the assistance of a smart wearable cast that incorporates a similar pattern of coils that automatically tighten with a servo motor based on the pressure surrounding the arm, the healthcare industry could shift completely with the advancement of a casting process. Instead of having to mechanically tighten a knob for the cast or cast, the smart device will establish pressure boundaries to automatically know how much to tighten on its own.

2.3 Sensors, Controllers, and Electromechanical Equipment

The main computer microcontroller that makes the smart cast function is the Arduino. Microcontrollers were first developed by 2 engineers named Gary Boone and Michael Cochran in 1971 and are devices that can be programmed, erased, and reprogrammed using only electrical signals (Wright Jr., 2022). A software application called an integrated development environment (IDE) normally consists of a source code editor, automation tools, and a debugger that can operate alongside a microcontroller and program to facilitate certain functions. Arduino is a brand known for their open-source hardware and software enterprise, initiative, and community of users. It focuses on crafting and producing single-board microcontrollers and microcontroller kits tailored for constructing digital devices. Arduino started developing microcontrollers with a mission to create an accessible and affordable electronics development platform in 2007. Arduino released their first successful and most popular microcontroller, the Arduino Uno, in 2010 and quickly became "the primary platform for makers, engineers and creators around the world" (Arduino Team, 2021). Through its evolution, Arduino can make hardware that adapts to specific needs based on its set of instructions making it very user-friendly and programmable for the smart cast project.

A pulse oximeter is a sensor device that will be used to measure the patient's heart rate and adjust the overall tightness within the cast. Pulse oximeters were first introduced in 1974 and have since become a tool for every doctor whether they are an orthopedist, cardiologist, or even a primary care doctor. Pulse oximeters measure the saturation of oxygen in the blood. They work by placing a probe with two LEDs on one side of a thin part of the body, such as an earlobe or finger, and a photoreceptor on the other (Hafen and Sharma, 2022). The probe emits light at 660nm (red) and 940nm (infrared) wavelengths on set intervals and the photoreceptor records the light absorbance (Jubran, 1999). The ratio of red to infrared light (red/infrared) is then placed into an algorithm that is programmed into the oximeter and the oxyhemoglobin saturation is displayed (Hafen and Sharma, 2022). Originally these sensors were clipped to the patient's ears and the processor was a big analog device that sat on the table. Today, the sensors and processors can fit into a small unit that can fit in the palm of one's hand and they are simply placed on a finger to monitor one’s oxyhemoglobin saturation. Although they are much less accurate, there are also wrist-mounted oximeters. These oximeters are much more limited due to the thickness and location of the LEDs. Smartwatches have integrated these wrist-mounted sensors in them to monitor the pulse of the user. A wrist-mounted oximeter will be integrated into the cast with the same purpose of monitoring the general pulse in the wrist post-surgery. A detachable finger clip sensor will also be included for more accurate measurements. Monitoring pulse and oxygen saturation is very important as the ulnar and radial arteries both run through the wrist and could need repair due to a traumatic injury. In the smart wearable cast project, a pulse oximeter could transmit data to the patient and correctly measure the heart rate of the patient.

Another important device utilized for the development of a smart arm cast design is a strain gauge pressure sensor. Pressure sensors function to measure, manage, and monitor pressure changes in engineering fields such as biomedical, aerospace, and the development of automobiles (Jena & Gupta, 2021). The invention of the strain gauge pressure transducer dates back to the late 1930s and has proven remarkable in engineering (Measurements Group, Inc., 1988). In the application of the smart wearable cast project, two strain gauge pressure sensors are used to detect the force when a person's arm comes into contact with the cast and help establish a baseline pressure reading. These signals will be communicated to the Arduino which will help engage an appropriate tightness to the user.

Another major mechanical component of the project is the incorporation of a small servo motor. Servo motors are self-contained electric devices that rotate or push parts of a machine with great precision control of angular or linear position, velocity, and acceleration in a mechanical system. Servo motors were first invented in the early 1960s, and since then have been new and approved over the years. Based on the given information about the servo motor, it was decided that a small servo motor that is compatible with the Arduino will be used because it will then be easier to control the shaft position of the arm cast very precisely with a high torque to inertia ratio allowing for there to be more accurate movement in the arm cast.

3.0 Methodology

This section aims to identify the procedures taken to create the initial prototype of the smart wearable cast device. The goal of this project is to design and develop a smart wearable arm cast device through the use of Mechanical Engineering principles and biomedical applications that will allow for arm casting to be less difficult/challenging. This device will have the potential to make a breakthrough in the healthcare industry with the use of sensor technology that is currently lacking in existence within this field.

The main objectives of the project are to:

1. Ideate a potential product that serves as a smart wearable device for consumers
2. Design a model with the use of mechanical equipment and engineering principles
3. Develop the model to create a functional prototype of the smart wearable cast
4. Test, analyze results, redesign, and redevelop the product for improvement

3.1 Cast Layering & Materials

Background research was conducted to study the current casting process including how it is assembled and taken apart as well as the materials that are incorporated in it. It was decided to use a similar design to the current plaster casts as well as add features and incorporate the use of a different material that has protective and flexible properties to best fit a patient's needs. This is because the smart cast design was created to have two layers, an inner layer made of a cotton sock and an outer shell that will tighten and loosen around the arm of the patient. The material needs to have the ability to compress and loosen around a person's arm as well as be durable to protect the arm from external sources. Thermoplastic polyurethane (TPU) is best suited considering the required material qualities are durability, light weight, flexibility, and rigidity, as well as being highly comparable to the mechanical and material properties of Plaster of Paris. This is shown in the tables below.

Thermoplastic Polyurethane Material Properties		
Property	**Value**	**Description**
Density [g/cc]	1.45	Light-weight, Low density
Hardness [R]	66.3	Durable, Rigidity
Max Tensile Strength [MPa]	63.3	Durable; Can withstand high force and pressure
Elastic Modulus [GPa]	2.58	Relatively low elastic modulus, easily stretches, does not deform
Elongation at Yield [%]	23.8	Amount material can stretch from its original state without breaking
Elongation at Break [%]	26.6	Amount material can stretch till breaking
Flexural Modulus [GPa]	2.18	Relatively flexible; Low resistance to bending force
Water Absorption [%]	0.24%	Low water absorption

Table 1: TPU Material Properties

Plaster of Paris Material Properties		
Property	**Value**	**Description**
Density [g/cc]	1.673	Light-weight, Low density
Hardness [R]	74.7	Durable, Hard
Tensile Strength [Mpa]	6.03	Low durability; Low resistance to pulling and stretching forces
Elastic Modulus [Gpa]	0.443	Low elastic modulus, easily stretches
Elongation at Tensile Strength [%]	3.25	Ability of material to deform before it fails under tension
Flexural Modulus [Gpa]	0.5	Flexible; Low resistance to bending force
Water Absorption [%]	29.50%	Adequate water absorption

Table 2: Plaster of Paris Material Properties

The main ideas considered for an appropriate material that would best support the project and the injured patient's needs were that the material must be strong and hard, yet have an ability to be flexible and elastic so it can compress and decompress when the tension force of the fish wire string is applied. TPU has not only proved to suit all these needs but has also been used for products across a multitude of industries including sports equipment, medical devices, and engineering (Xu et al., 2020). Moreover, TPU has a unique ability to be used for 3D printing applications with its filament form, making it an efficient tool and resource to design and produce the main body of the cast using computer-controlled (CNC) machinery. Through continuous design adjustments and improvements, a functional cast prototype made from the TPU material was created.

3.1.1 TPU Shell

The main cast structure used in our final prototype was first created in the computer-aided design (CAD) software SolidWorks. Measurements of the diameter of a team member's wrist and cross section just below the elbow were taken to create an overlapping cross-section on the computer software. A lofted feature helped maintain the expanding shape of the cast design so it could wrap around a patient's arm efficiently. The cross-sectional wrist diameter of the cast body is 54 millimeters and the cross-sectional elbow diameter is approximately 97 millimeters. Due to 3D printing requirements on the Prusa i3 MK2S, the original length of the cast structure was reduced to 6 inches. The initial intention of this layer was to protect the patient's arm from the electromechanical equipment being used, thus having slots holding the equipment and a plan for an external third layer that will attach to the second layer and cover the devices being used. This design idea had several flaws including the necessity to make the second layer act as the layer that tightens and loosens due to holding the servo motor and pulley, as well as the incorporation of lofted walls that will help feed the nylon wires through line holes. These two exterior walls have heights of 13 millimeters and 5 millimeters consisting of a series of seven 5-millimeter holes on one side and eight 5-millimeter holes on the opposite wall. The model had the alignment of the holes alternate from each other on opposite sides for the nylon string to alternate and weave through. The cast body was also carefully designed with a bulky and shelled-out bottom loft that will hold a 4.5 Volt battery power source, an Arduino Nano R3 board that we will be using to control the sensors and motor, and a mini breadboard for wiring. Lastly, for the basic shape of the cast body structure, measurements of the Tower Pro SG92R servo motor were taken to create an initial slot along the curved side of the cast. These dimensions of the motor slot are 13.5 millimeters wide and 18 millimeters long. The first cast shell was then created and ready to be initially printed out of TPU as seen in the figure below.

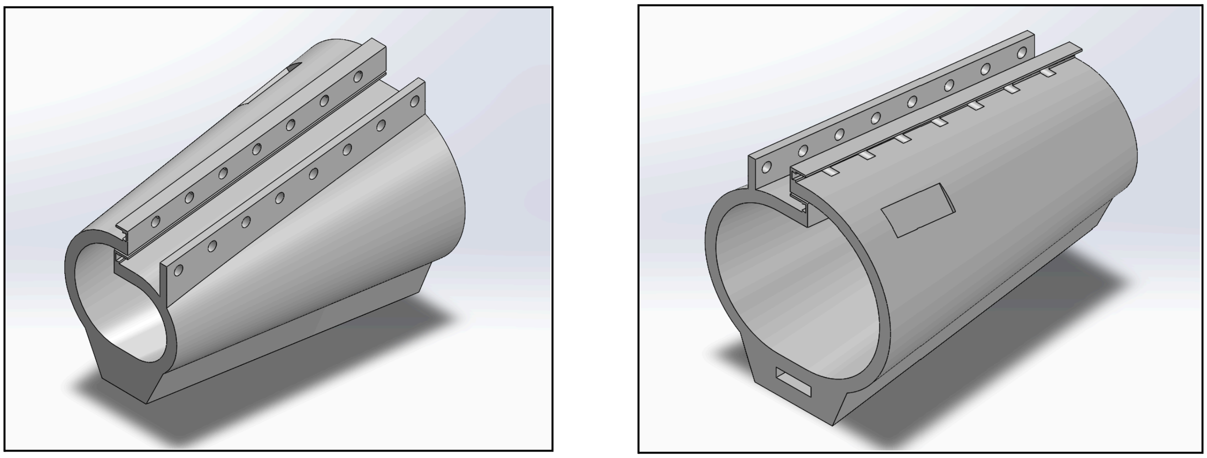


Figure 2: Isometric and Back-Right View of Early Stage of Cast Second Layer

3.1.2 Cotton insulation

Typical plaster casts utilize cotton on the inside to protect a patient's skin and provide additional comfort, the inner layer design was made from a cotton arm sock material to act as a cushion directly around a person's arm for comfort and protection from the rough TPU shell material. The inner sock was cut to match the 6-inch length of the arm cast and applied using a combination of gorilla glue spray and gel adhesive to stick to the TPU cast layer. Additionally, the strain gauge pressure sensors sit within this layer, thus making it important to pass more cotton fabric to cover the wires so that the sensors make direct contact with the skin without irritating the patient.

3.2 Computations for Arm Cast

Due to the manner of this project incorporating the use of electrical equipment, a central brain needs to act as a middleman for sensory inputs and mechanical outputs. To accomplish this function, an Arduino Nano R3 was used to coordinate the sensory components and tightening mechanism, described later in this Methodology.

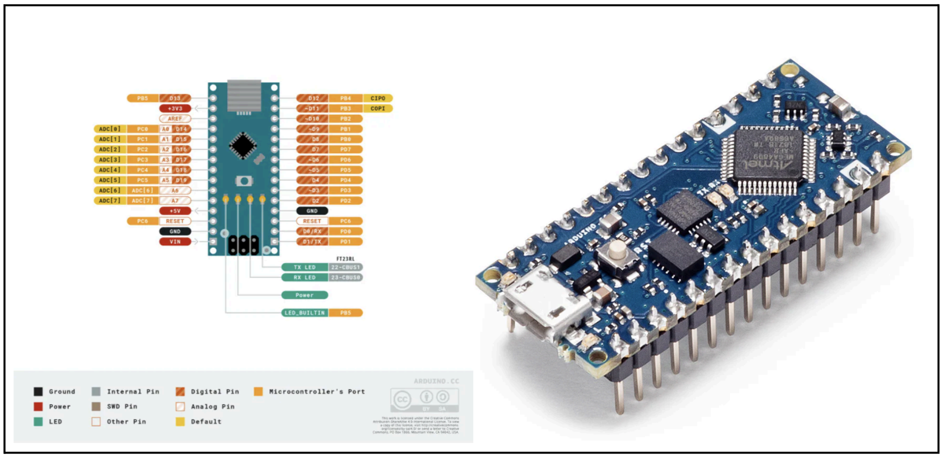


Figure 3: Image of Arduino Nano R3 Used in Arm Cast Alongside Pinout Diagram

The Arduino Nano r3 uses a series of digital and analog pins for either input or output pins as well as general supply power and ground pinouts for common circuitry. Pins A4 and A5 are special pinouts, acting as the Serial Data Line (SDA) and Specification and Description Language (SDL), and are used for specific data collection and for one of the sensors below. For the purpose of the electrical equipment in this arm cast 4 analog pins for inputs (A0, A1, A4, A5), 1 digital pin for input (6), 7 digital pins for output (2, 3, 4, 5, 9, 11, 12), ground pin and 3.3V pins are used.

3.3: Sensory & Display Components

In order to calculate the needed pressures and servo movements to tighten the arm cast, a series of sensors need to be placed across the cast. Along with this, the data collected from these sensors needs to be displayed for the patient in order to have a grasp of what the arm cast is doing for them. For the inputs into the Arduino, the sensors used consist of two strain gauges and a single MAX30102 Pulse Oximeter. For the patient to activate specific phases of the cast process, such as initial calibration and recalibration of tightness, a switch button is used. For the data being displayed, a 16x2 LCD is used. The following sections describe each specific piece of equipment, its purpose in the cast, what data it either sends to or receives from the Arduino, and where it is placed on the cast.

3.3.1 Measuring of Cast Compression

Figure 4: Image of the Strain Gauge Model used in the Arm Cast

The most important sensors used by the Arduino are the two strain gauge sensors, whose purpose is to measure the mean pressure placed on the patient's arm by the arm cast. The chosen strain gauge model, the Flexiforce A201-1 has a diameter of 9.52 mm and functions by converting pressure placed upon it and converts it to a gram force value in a range from 0-453.6 grams force (1 gram force is equal to 9.8E-5 N). For the purpose of this project, the assumed constant pressure for a patient with a broken arm is around 1.02 Newtons, placed on the patient (Tuan et al.). On the arm cast, the strain gauges are placed inside the shell of the cast at the horizontal midpoints and on either side. This location was chosen to make sure that these sensors could take readings of the pressure of the cast at different points around the arm for the purpose of measuring an average pressure. This average is calculated inside the Arduino after taking the strain readings from the gauges and then mapping it to a bit scale (from 0-453.6 to 0-1023). This value is then mapped once more as a variable for the Arduino to compare to the degree value the servo motor is set at. This activation of the stain gauges occurs at two stages of the arm cast process: when the cast is first activated, and when the patient presses the button on the cast. The initial activation occurs in order to establish the required tightness needed to be maintained for the patient in order to receive proper pressure for arm recovery. This input is stored in the Arduino and is referred to at multiple points in the Arduino's code. However, if needed to be changed, the staging gauges can be activated again by pressing the button after the cast's initial start-up to compare the current tightness to the initial value determined. From this comparison, the cast can readjust the tightness via the servo motor so that the pressure returns to its original value.

3.3.2 Pulse Oximetry

Figure 5: Image of the MAX30102 Pulse Oximeter used in the Arm Cast

Acting as a secondary means of determining the tightness value of the arm cast, the pulse oximeter is used to determine the average heart rate of the patient at a given time. The pulse oximeter functions by lighting an LED that reflects on the blood flowing in a vessel of a patient and back on a photodiode. The comparison of the wavelength of the light sent by the LED before and after reflecting off the patient's blood is able to measure how fast the blood is flowing, and thereby the patient's heart rate. On the arm cast, the pulse oximeter is placed below the wrist of the patient, attached to the outer front face of the cast. The purpose of using the pulse oximeter is to detect when the patient’s arm is in the process of swelling or contracting due to the lack of blood flowing in the (Huggenberger & Detmar, 2011). When active, the pulse oximeter can determine the heart rate of the patient, thereby the extent of how expanded the blood vessels in the arm are. From this data, the Arduino communicates to the servo motor to adjust its positioning to tighten a patient’s cast at low heart rate states, such as <60 BPM, during a period of low activity or rest. On the opposite end, if the patient has a high heart rate, such as >120 BPM, during a period of high activity or exercise.

3.3.3 Displaying of Data on Cast

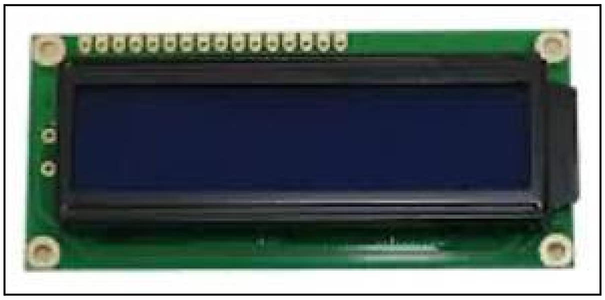


Figure 6: Image of LCD Device

Lastly, with the collection of the sensor data, a Liquid Crystal Display (LCD) is installed on the upper left face of the outer cast. The LCD takes data from the Arduino and displays text on a 16x2 digital board. The purpose of installing an LCD to the cast is to give the patient a window into seeing some of the digital data collected by the sensors. During the initial start-up of the cast, the LCD shows a series of texts that help explain what the patient can do with the cast. The text “Press Button to Set Tightness” is displayed at the beginning, signaling the patient to set their initial tightness value. After the button is pressed, activating the strain gauges, the text “Servo value set from 0-180” is displayed, helping the patient understand the servo variable data. After initial startup, the LCD moves to display the two main variables collected by the cast’s sensors. The first line of the LCD displays the collected average heart rate of the patient, while the second line shows the current position set at the servo motor. Allowing the patient to see their heart rate along with the current servo positioning

3.4: Tightening Mechanism Overview

In order for the cast to tighten and loosen, the cast has a tightening mechanism made up of a servo motor, a custom-made pulley wheel, and nylon wire fed through holes in the cast. The tightening process begins with the Arduino Nano R3 sending an electrical signal to the servo motor, causing its output gear to turn either clockwise or counterclockwise. The custom-made pulley is attached to the output gear of the servo motor, so it turns with the rotation of the output gear. The pulley is connected to both ends of the nylon wire that runs through holes in the main body of the cast so when the pulley turns, the two sides of the cast are pulled together or released to move apart. This entire tightening process can be triggered by pressing the button, located on the base of the thumb. When the button is pressed, the Arduino Nano R3 compares the current sensor readings to the established baseline and sends a signal to the servo to adjust the diameter of the cast so they match. Then, the baseline measurements are reestablished.

3.4.1 Servo Motor

Figure 7: Image of Tower Pro SG92R Mini Servo Motor

As previously mentioned, the first part of the tightening mechanism is the servo motor. The servo motor used in this cast is a Tower Pro SG92R, shown above in Figure X. This servo motor is a DC motor connected to a potentiometer to keep track of the distance the output shaft has turned. The servo motor has an operating voltage of 4.8 Volts and rotates at a top speed of 600 degrees/s. The motor was chosen because it has a gearbox so the max output torque of the motor is 24.517 N-cm. This motor's max height (from base to top of the drive shaft) is 34.5mm, which makes it one of the smallest servo motors on the market, and another reason why this motor was chosen. The last contributing factor to this motor is that it's low in price, only costing about six dollars (Tower Pro, 2024).

3.4.2 Pulley Wheel

The second part of the tightening mechanism is the pulley wheel. The pulley wheel was completely designed and fabricated by the team. The pulley was designed with a base circle sketch that has a radius of 18 mm and was extruded 6.4 mm in height. These dimensions were chosen to neatly hide the spooled-up wire and keep the overall pulley low profile. For determining the hole radius of the pulley wheel, 4 different files of pulley wheels with the exact same size were created with different hole diameter dimensions to tolerance test fit each one on the servo motor. The servo motor shaft diameter was measured at 4.75 millimeters using calipers. In SolidWorks, the 4 identical pulley wheels had hole diameters of 4.75, 4.85, 4.95, and 5 millimeters.

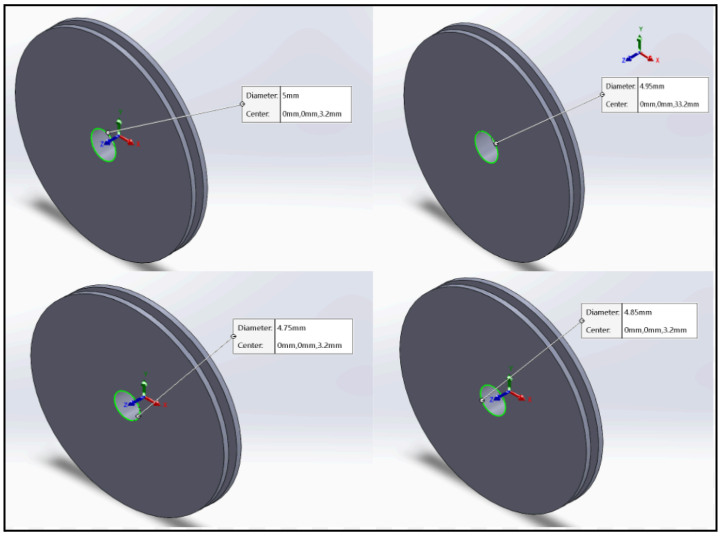


Figure 8: Tolerance Tested Pulley Wheels in SolidWorks

These files were then printed using PLA filament and press fit on the servo motor shaft to determine the most optimal wheel. The final pulley wheel has a hole in the middle of it with a diameter of 4.95 mm. This pulley wheel has a snug fit and the ability to rotate around the output shaft of the servo motor. The smaller outer radius for the location of spooled wire is 15.5 mm and the width of the cut is 3 mm. The outer radius dimension was chosen so the wire is brought in at a rate of approximately 49 millimeters per rotation. The width of the cut was chosen to ensure that the line spools up easily.

3.4.3 Nylon Wire

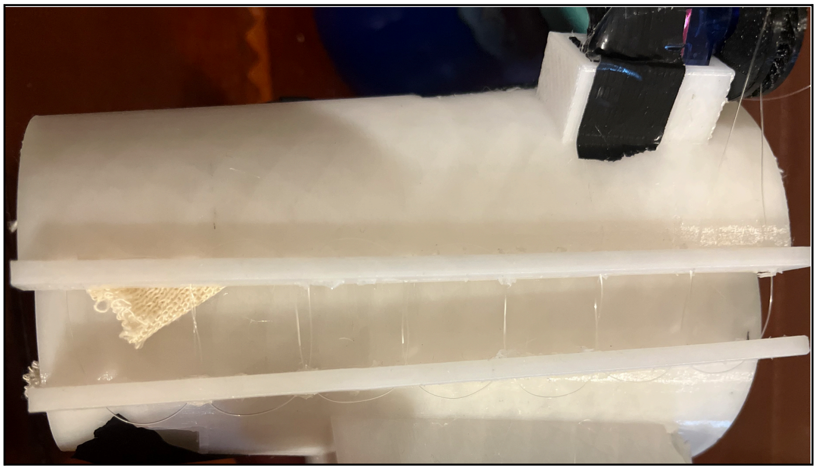


Figure 9: Nylon Wire Fed Through the Splint

The final part of the tightening mechanism is the threaded nylon wire. The wire is threaded straight across (from the top to the bottom side of Figure X) through the two holes at the right (near the elbow) of the splint. Then, the wire is threaded back across (bottom to top) through the two holes to the left of the farthest right two holes. This threading pattern is continued all the way to the left of the splint (near the wrist) and back to the two holes to the left of the far right two holes. There are a total of eight holes on each side that have an equal distance of 15 mm between the edge of each circle. This thread pattern allows for an evenly distributed force across each section of the wall on each side of the cast. The evenly distributed force on each wall results in an even and consistent tightening across the entire length of the cast when the wire is pulled in.

4.0: Assembly & Testing

This section of the report outlines the final results ranging from the assembly created from the final CAD designs to overall testing of the sensor equipment, tightening mechanism, code calibration, and performance testing of the assembled smart cast. These tests help indicate the functionality of the short arm cast as well as supply analytical data from the culmination of Arduino code and the sensors used. This data was compiled and analyzed to support claims and objectives for future work and the longevity of the smart cast.

4.1 Final Cast Model Design (CAD)

The final cast design incorporated adjustments to all slots holding mechanical and electrical equipment. The slot for the motor was kept with a surrounding 3-millimeter thick casing added to help keep the device in place. A 4 by 10-millimeter rectangular slot for the wiring to go through was also incorporated to make a connection with the Arduino. Due to the design of the cast being made for a patient's right arm, another case with a width of 33.5 millimeters and length of 78.1 millimeters was created on the back left side of the cast main body to hold the liquid crystal display (LCD). This will allow the patient to easily view their current biometric data with the heart rate in beats per minute and overall pressure that the system has around their arm. This case has a 3-millimeter thick wall and a slot depth of 32 millimeters, thus being able to fit the wiring underneath and through a 4.5 by 72-millimeter window created on the side of the casing. The bottom loft also needed to have a size increase to be able to fit the 3 pieces of equipment underneath the cast. The bottom loft at the wrist and elbow have a height of 30 and 45 millimeters respectively. The final iteration of the cast body was changing the alignment of the holes to be directly across the opposing wall side and the additional hole to make an equal series of 8 holes on each tightening wall. Once the final main cast structure was created in SolidWorks, an assembly was made using CAD models of each component used in the smart cast to show where all pieces of equipment were sitting and create a visual for what the real-life model would look like as shown in the figures below.

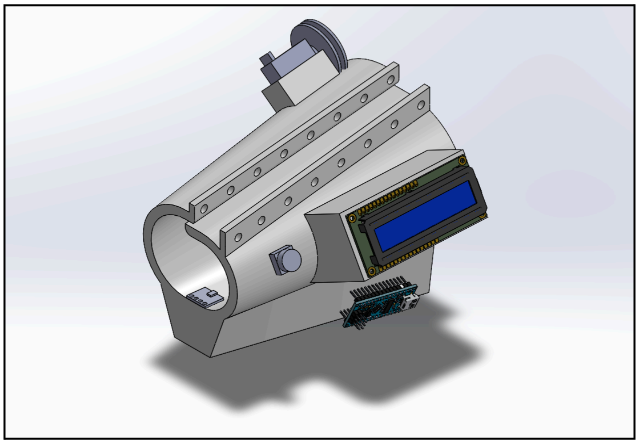


Figure 10: SolidWorks Assembly of Smart Arm Cast

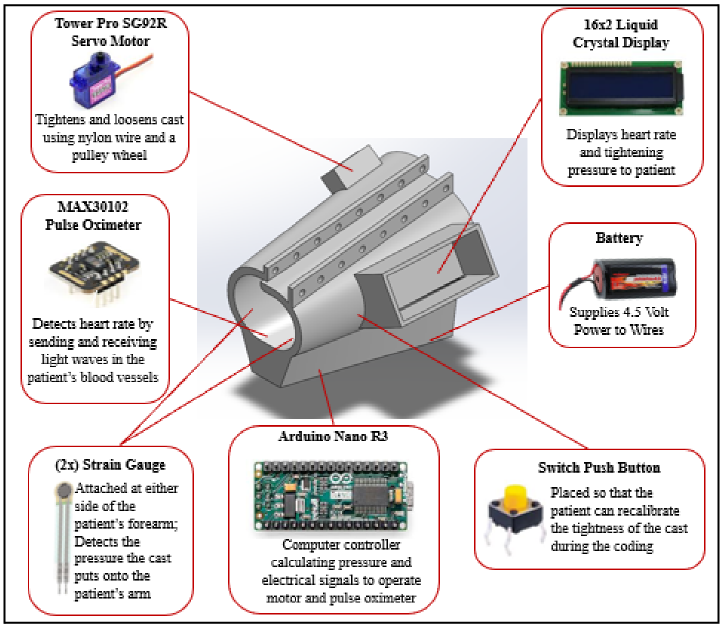


Figure 11: Labeled Diagram of Each Sensor with the Cast CAD Model

4.2 Combined Arduino Circuitry

To make sure that the wiring was able to be performed correctly, the combination of all sensors, motors, and wiring accessories to the Arduino was first assembled in Fritzing, a software that allows for digital Arduino wiring. The following subsections describe the final design of the circuitry and what was included in it, along with an explanation of the code used to perform the sensor collection and servo tightening

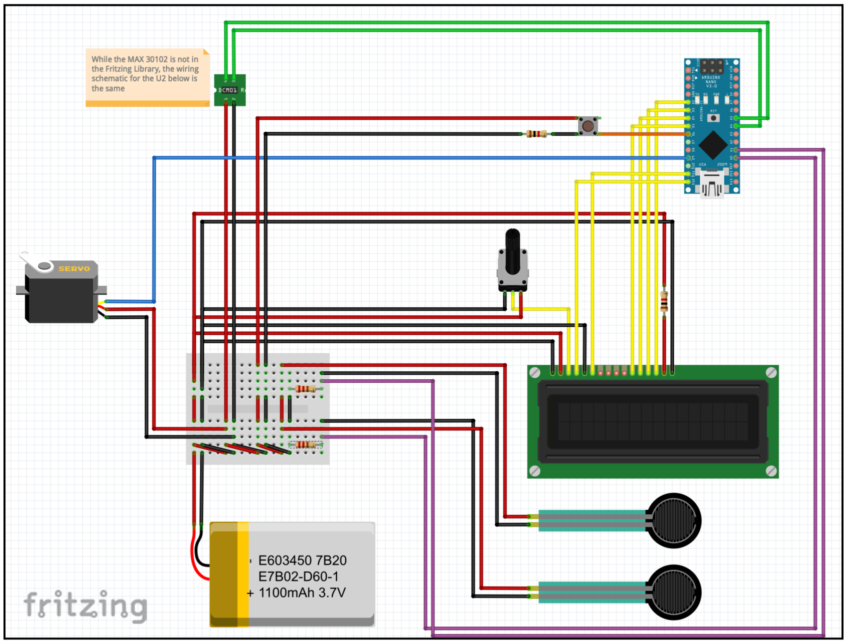


Figure 12: Fritzing Digital Diagram of Complete Arduino Circuit

4.2.1 Circuit Assembly

As seen in Figure 12, the circuitry of the Arduino and its accessories were assembled using a series of color-specific wiring, resistors, and a potentiometer. This was in order to make sure the circuit would not malfunction or receive enough electricity at one time to short or ruin the cast assembly. In order to have accurate readings from the strain gauge from the internal cast pressure, 220-ohm resistors were used to increase the base resistance of that section of the circuit. Two 1 kOhm resistor was used for the button and supply power for the LCD in order to not disrupt the digital pin or short the LCD out respectively. Lastly, a 10 kOhm potentiometer was used to change the light level of the backlight of the LCD. This was the only piece of circuitry equipment that did not contribute to the casting code that was accessible to the patient, allowing them to adjust the brightness of the LCD. The diagram also used specific wiring colors to classify the purpose of each of the wires. Red and black wiring represented either the 3.3V supply or ground for each of the pieces of equipment. Yellow wiring was directed to the LCD, green to the pulse oximeter, blue to the servo motor, orange to the button, and purple to the strain gauges.

4.2.2 Flow of Code

For an analysis of the code that is used for the tightening and data collection of the cast, the code was split into three specific phases for an easier form of organization: Initial Phase, Passive Phase, and Active Phase. During the Initial Phase of the cast code, the sensors are activated, the servo motor is set at 90 degrees, the midpoint value of the servo positioning that can also provide the 1.02 N of needed pressure, and the LCD is activated. The patient is told by the LCD to press the button to assign the tightness value that is wanted to be maintained by the servo motor. After pressing the button, the arduino records the strain gauge values, takes their average, and sets the new baseline pressure. Should the pressure of the cast be higher or lower than the needed 1.02 N pressure, another variable is generated, acting as a cushion variable for the patient, which is added to the measurements value for the remainder of the code.

After assigning the needed maintain servo positioning and baseline and cushion pressure variables, the arm cast enters the Passive Phase of the code. This part of the code is planned to loop indefinitely and is also returned to after the code is performed in the Active Phase. During this phase, the pulse oximeter takes readings from the patient’s wrist and calculates an average heart rate. From this value, the arduino calculates whenever or not to adjust the servo motor. If the heart rate is measured under 60BPM, signifying a low blood flow and contraction of the arm’s blood vessels, the servo motor is rotated counterclockwise by 5 degrees from the original position. This causes the tightening wire to retract, tightening the overall cast pressure on the patient by about 0.2 N (Tuan et al.). On the other hand, if the patient’s heart rate is over 120 BPM, the servo motor is rotated clockwise by 5 degrees from the original position, tightening the overall cast pressure on the patient by 0.2 N. These movements of the servo motor do not interfere with the baseline pressure value, only adding or subtracting from the servo positioning assignment by it. Should the patient remain in the normal range of 60-120 BPM, the servo will not adjust and optimally remain at the original pressure. After this heart rate comparison, it and the current servo positioning is displayed on the LCD.

Lastly, should the tightness of the cast become too tight or too loose, either through natural creep of the wiring or a hit on the cast itself, the patient can press the button during the Passive Phase of the code in order to activate the Active Phase of the code. During the Active Phase, the strain gauges take another reading of the current pressure and compare it to the original baseline pressure variable with the cushion variable applied. Should the pressure deviate too far from the 1.02 N baseline pressure from the pressure and cushion variable, the servo motor will adjust its position so the strain gauge measures the baseline again. Because the servo needs to be changed for this adjustment, the LCD will update and show the current positioning.

4.3 Final Smart Arm Cast Assembly

The final assembly of the arm cast was created after the main cast body structure was printed after 1 day and 8 hours. The final model consists of 3 main slots, one for holding the servo motor in place, one for holding the LCD, and a bulky, shelled-out bottom loft for holding the battery power source, the Arduino Nano R3, and the mini breadboard. The final cast model also included adjusted and symmetrical holes on each tightening wall ensuring even tightening across the top of the cast. The inner layer of cotton sock was applied on the inside of the TPU shell using a combination of Gorilla Glue spray and and Gorilla Glue clear moisture adhesive. The adhesives were applied on the entire inner portion of the TPU cast while being stretched open and pressed for approximately 1 minute throughout. Excess cotton was trimmed to the length of the cast and saved to cover up the wiring from the strain gauge pressure sensors. Next, all wires needed were connected to each device used and fed through their respective slots. The windows on each case would help each piece of equipment be able to establish a connection to one another while also being able to comfortably sit in place. After several iterations of dimensions on the TPU main body, each piece of equipment was able to fit freely in their respective slots. To reduce excess wire length, the team used a soldering kit to first cut a portion of the wires and then join the metal portions together through its heating process. Black tape was used throughout the cast to make sure wiring and specific pieces of equipment did not detach from the cast. Once this was completed, all the electromechanical equipment was combined with the TPU cast and cotton layer and testing was ready to be conducted as shown in the figure below.

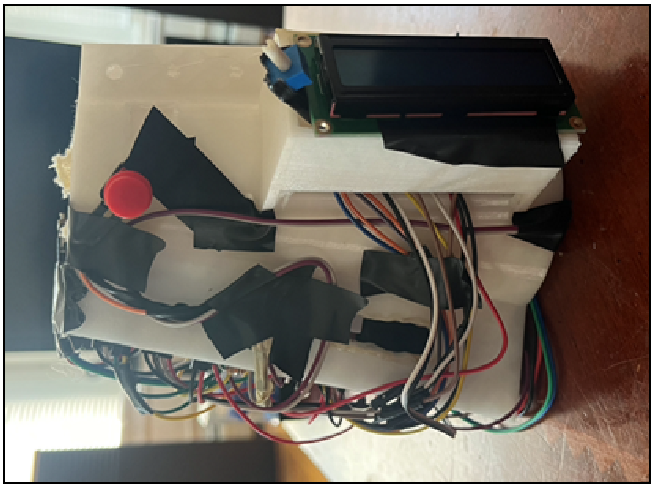


Figure 13: Photograph of Assembled Smart Arm Cast with Wires

4.4 Project Testing

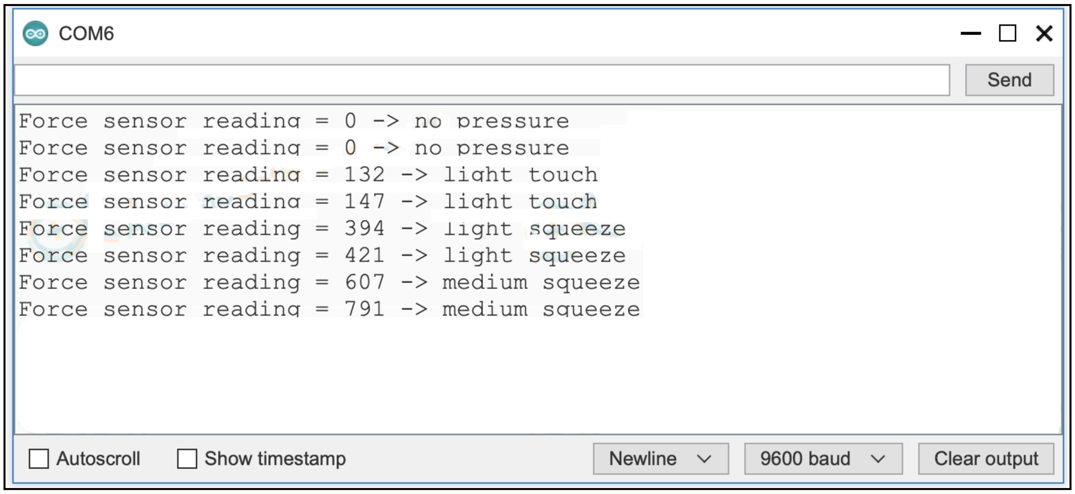
The testing performed on the cast can be broken down into three main parts. The first part of the testing was the wire tension testing. The wire tension test was done to measure the tension force on the nylon wire as it was pulled to decrease the diameter of the cast. The second part of the testing was the pre-code testing. This was performed to ensure the pressure sensors, pulse oximeter, Arduino Nano R3, and servo motor all work independently of each other. The third, and final, part of the testing was the code testing. The code testing was done in three phases. The first phase was testing the code for Arduino Nano R3 to translate the pressure sensor data to servo motor movements. The second phase of the code testing is the Arduino Nano R3 to translate the pulse oximetry data into servo motor movements. The final phase was combining the two previous sets of code when the button was pushed.

4.4.1 Wire Tension Testing

The nylon wire was used to run through the wall of the interchanging holes that were created to pull the cast together. When measuring the wire tension, we used a device called the Mecmesin Advanced Force Gauge (AFG). The AFG is a handheld digital force gauge used for tension and testing. As the team began testing the strength of the wire we first made sure that the nylon wire was properly fed through the holes built into the cast, and calibrated the sensor with a 250 Newton weight. Secondly, we tied the open end of the nylon wire to the hook that's on the AFG device and then ensured to zero out the device so that we didn't receive an incorrect reading. Afterward, we gave a constant even pull on the cast as the wire was attached to the hook to get a reading of about 8 Newtons. This testing was very important because we were then able to understand the cast fixation, whether it meets the expectation for functionality, and whether it can effectively support the injured arm during the healing process.

4.4.2 Sensor Calibration

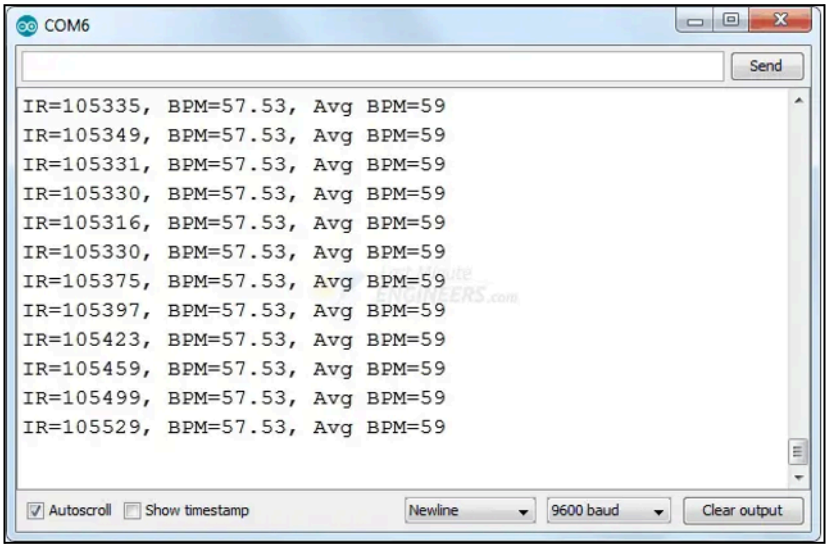
After placing strain gauges and Pulse Oximeter on the cast, they needed to be tested in order to make sure they are able to detect proper measurements. This is started with the force sensors, where an individual’s arm is placed in the cast and the cast is stretched. The code used in this test is a piece of the final code, however only using the strain gauges and arduino nano. After testing the range of the strain gauge sensors, a pressure of 1.2 N and 1.4 N of force. In order to make sure the pressure sensors are being adjusted, the Arduino Serial Monitor is opened on an adjacent computer supporting Arduino.ide software. On the Serial Monitor, the expected byte values are stored around the 790-810 range.



```
Force sensor reading = 0 -> no pressure
Force sensor reading = 0 -> no pressure
Force sensor reading = 132 -> light touch
Force sensor reading = 147 -> light touch
Force sensor reading = 394 -> light squeeze
Force sensor reading = 421 -> light squeeze
Force sensor reading = 607 -> medium squeeze
Force sensor reading = 791 -> medium squeeze
```

Figure 14: Force Sensor Testing and Calibration shown by Serial Monitor

After calibration of the force sensors, the pulse oximeter needed to be tested separately from the other sensors. This was done by performing a similar test of the strain gauges, where a piece of the final code using only the pulse oximeter was isolated and looped. This allowed for the heart rate to be constantly collected and able to be calibrated in order to determine if the sensor was capable of working in the final arduino circuit.



```
IR=105335, BPM=57.53, Avg BPM=59
IR=105349, BPM=57.53, Avg BPM=59
IR=105331, BPM=57.53, Avg BPM=59
IR=105330, BPM=57.53, Avg BPM=59
IR=105316, BPM=57.53, Avg BPM=59
IR=105330, BPM=57.53, Avg BPM=59
IR=105375, BPM=57.53, Avg BPM=59
IR=105397, BPM=57.53, Avg BPM=59
IR=105423, BPM=57.53, Avg BPM=59
IR=105459, BPM=57.53, Avg BPM=59
IR=105499, BPM=57.53, Avg BPM=59
IR=105529, BPM=57.53, Avg BPM=59
```

Figure 15: Serial Monitor of the Pulse Oximeter

Lastly, the LCD code from the final casting code was isolated in order to make sure that the LCD could display text and changing data over time. Because of the lack of sensors for the LCD, the main concern for the device was to make sure that the output data was not being changed or disrupted by the wires or LCD itself, and that it was able to fit in the available 16x2 space.

4.4.3 Baseline Tightness Testing

After making sure that the sensors, LCD, and servo motor are able to respond to the arduino command, the Initial Phase was tested in order to see if the baseline variables were able to be saved by the arduino and effectively maintain the baseline servo positioning over an extended period of time. For the duration of 30 minutes after establishing the baseline variable and cushion variable, the cast is maintained and then tested to see if the value would deviate, which would be assumed to be from the tightening wire loosening.

4.4.4 Tightness Adjustment from Heart Rate

Following the testing of the Initial Phase, the code used for the Passive Phase was selected to be used from the final code. Specifically, over the course of an hour, the pulse oximeter would measure the heart rate of the test subject in the arm, and the servo motor would either tighten or loosen if the heart rate was less than 60 BPM or above 120 BPM respectively. The goal of this testing process is to make sure that the arduino will keep the baseline value of the tightness after temporarily increasing or decreasing because of their heart rate.

4.4.5 Tightening Recalibration

The final test is to perform the Active Phase of the arduino code. This is performed by physically shifting the pulley wheel when removed from the servo motor, ensuring the servo positioning does not change in the process. After this, the button on the arm cast is pressed, activating the strain gauges and comparing the two values and readjusting so the tightness is returned to the baseline.

5.0 Results and Findings

This chapter aims to specify all of the observations and analytical results that were found after testing the smart wearable cast. Specific results found were based on the tension of the wire-pulley system and the biometric and pressure data associated with it when someone is wearing the cast. This data proved helpful in the functionality of the smart cast while also giving insight into what could be improved and added in the future.

5.1 Wire Tension Findings

When measuring the wire tension, the team realized that based on the automatic tension gauge (ATG) reading the tension peaked at 8 Newtons. As the team measured the diameter of the servo motor spindle the team then determined what the diameter of the pulley should be which was 4.95 mm which would give us an accurate range of motion and simulation in the wire tension. The pulley wheel was then placed on the servo motor, and wrapped with the nylon wire to show an accurate effect on the force transmission in the smart arm cast system where it was determined that because the diameter measurement was a small and reasonable amount of force was used to pull the wire it allowed for there to be high tension in the arm cast. Therefore, it was important that the team got the right measurement reading from the ATG because it told us the minimum diameter that the pulley wheels' maximum radius needed to be in order to understand the fixation of the arm cast.

5.2 Results from Smart Arm Cast Testing

When the initial calibration of the arduino equipment on the arm cast testing was performed, both types of sensors were able to work accordingly. While the strain gauges were able to effectively establish a baseline pressure reading, the communication of signals between the sensors and Arduino proved to have a long delay, thus affecting the speed the tightening mechanism is able to perform. The sensors are located at both sides of the wrist and have not been tested in other locations. Furthermore, in previous studies, an increase in the amount of strain gauges shown to have an increase in the pressure accuracy (Tuan et. al). When testing the pulse oximeter, while the initial readings were unable to be determined, after around 30 seconds of direct placement on the wrist, the pulse oximeter was able to begin collecting and sending heart rate data to the arduino. Collection of the patient’s heart rate at a specific instance took around 10-12 seconds to collect and send.

During the testing of the individual phases of the Cast Code, it was shown that the initial pressure baseline data was able to be recorded during the Initial Phase and be temporarily modified during the Passive Phase when the heart rate was outside the 60-120 BPM range. One of the issues presented with the collection of the BPM however was the plateauing of the average

BPM variable, as overtime the average would become harder to change since readings close to each other would collect overtime. This however can be adjusted with the code being updated to clear the cache of old BPM readings after a period of time, keeping only a limited amount of data readings from the pulse oximeter and deleting the oldest data each time a new reading is measured. Lastly, while the Active phase was able to return the arduino to the baseline pressure readings by adjusting the servo's positioning, the degree of repositioning over the course of an hour of compression was little. The average amount of change needed by the servo to return the pressure to baseline was 4 degrees counterclockwise every hour. This amount of change of the pressure can be amounted to the creep of the pulley wheel and tightening wire.

6.0 Future Works and Improvements

This chapter of the report serves as guidelines for a continuation of this project as well as improvements based on the analysis and findings based on tests conducted with the smart arm cast. Changes to the project are encouraged if this project were to be improved later on and serve to be crucial if the device were to be used on patients in the healthcare industry, These iterations include design changes and additions to the cast body, further testing on the long-term durability of the assembled cast and TPU material, overall pressure distribution improvements, coding improvements to minimize delay between the Arduino and connected devices, as well as the fabrication of a connected hand portion to the short arm cast to ensure complete fixation of the wrist.

6.1 Changes to Arm Cast Design

Although the smart cast body is able to securely hold each piece of electromechanical equipment, additions to this design could be made to improve its quality and cover the devices and wiring. An outer coating designed in CAD would ensure the protection of the equipment used while also protecting the patient through its additional thickness and saving them from the potential harm from exposed electronics. The first piece of covering that can be added is a bottom lid that would neatly fit in the slot beneath the cast, sealing off the area holding the Arduino, battery, and breadboard. The bottom of the cast has a lofted dovetail slot that symmetrically has an outer length of 6 millimeters, an inner length of 3 millimeters, and an overall width of 30.32 millimeters. The loft of the slot protrudes into the cast a total of 150 millimeters.

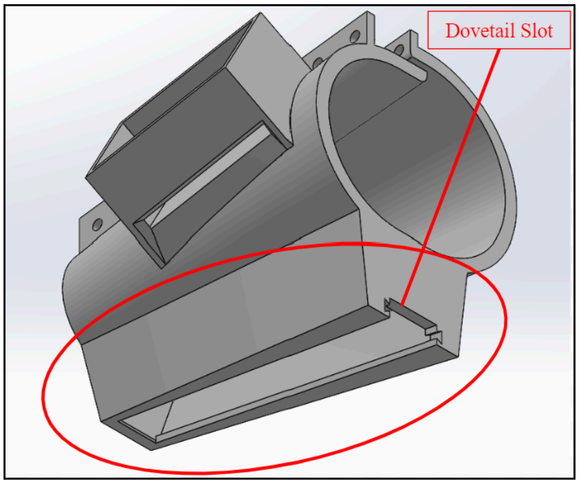


Figure 16: Picture Showing Area To Be Covered and Associated Dovetail Slot

Finally, an outer layer separate from the cast can be created to cover the wires from the button, LCD, and motor. To do this, an offset of the outer crust of the cast can be taken in SolidWorks that includes the outline of the elbow, wrist, and all casing from the front and back view of the cast. Once both profiles are made at both the wrist and elbow locations, the lofted feature can be used to create an expanding cover of the main cast body. Both the bottom lid and exterior cover can then be 3D printed using TPU filament.

6.2 Optimization of Tightness Distribution Around Cast

As mentioned in section 2 of the report, four of the analog lanes on the Arduino Nano R3 are used by the sensors. Of the four lanes in use, two of them are used by the pulse oximeter and the other two are used by the pressure gauge. Each pressure gauge uses one of the analog pins, meaning four more sensors could be added to the system. If the project were to continue, more of these sensors would have been added to better understand where the greatest points of pressure inside the cast are. This information would be incredibly helpful for understanding how the pressure at different points along the inside of the wrist and possible hand portions of the cast on the user's arm as the cast tightens and loosens.

6.3 Extension of Cast into the Hand

General arm casts and splints cover a portion of the hand up to the knuckle of the patient in order to adequately fixate the wrist. This is typically done through either the molding of casts around a patient's arm or the use of Velcro on splints. A proposed model for the hand portion of the cast was created in SolidWorks. To do this, an open-source CAD file of a human hand was utilized to be merged in a layer of extruded circles. Then the two bodies were combined and the result was one merged body in SolidWorks. Then the hand portion that was once merged could be subtracted using the "combine" feature in the CAD software to result in a mold that could fit fingers through it. Extruded cuts were made through the part so that the entire hand including the thumb could easily fit through it. Lastly, a sketch profile was created using the same cross-sectional wrist diameter dimensions that were used for the arm cast to create a piece that would fit perfectly with the printed cast body. This file could not be 3D printed due to the complex geometry of both the hand and the resulting shape of the model as well as its inability to be printed in one piece with the arm cast due to the combined height exceeding the limits of the printing bed. Overall, the hand portion of the cast was created and can be seen in the figure below.

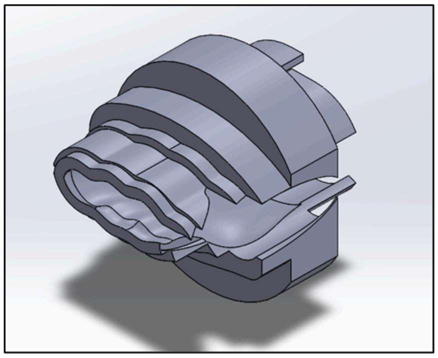


Figure 17: Proposed Hand Portion To Be Attached To Cast Body

6.4 Further Testing

For the continuation of the project in the future it is critical to acquire specific data on long term durability of the cast. One measurement important to consider is the creep factor of the tightening mechanism on the TPU material. Creep deformation measures the tendency of a solid material to undergo deformation after a long period of a time due to mechanical stress. The tightening mechanism constantly puts stress on the material when it engages the motor to achieve different pressure values within the cast. Steady state creep can be calculated using the equation:

$$\textbf{Creep Rate} = A\sigma^{n}\mathbf{exp}(-Q\mathbf{RT})$$

This equation shows that "A" is a constant dependent on the material and creep mechanism, "n" is an exponent dependent on the creep mechanism, "R" is the gas constant, and "Q" is the activation energy of the creep mechanism. Along with creep, further material testing is needed in order to justify that the TPU is protective and comparable to the use of plaster or fiberglass for casting protection.

Alongside further material testing, it is also important to consider long-term functionality of the smart cast. Testing when wearing the short arm cast typically had a duration of one hour of wearing before the cast was taken off. An injured patient would need constant fixation of the wrist for appropriate recovery and wears a cast for about 6-12 weeks. If possible further testing with patients who can wear the cast for longer periods of time will help provide more analytical data of the functionality and effectiveness.

Lastly, the smart electromechanical equipment can be further tested for optimal usage. There is a slight delay when the pressure sensors and Arduino communicate the signals to one another, which affects the speed that the motor to spin and tighten the cast appropriately. Changes to the code and sensor placement can improve the delay time making the cast more efficient to use overall. Additionally, there are unused pins in the Arduino controller used that allow for up to 4 more strain gauge being able to connect to it. This in turn, can establish a more appropriate baseline and more accurate pressure data within the cast that will also improve the effectiveness of the smart arm cast.

7.0 Broader Impacts

This chapter outlines the parameters and motivating elements of the project based on the impacts of people, culture, the environment, and economics. It is important to consider these factors which stress the significance of the product created and how it can be critical to the world we live in today based on the function, manufacturing, distribution, and any other relevant aspect of our smart wearable cast device.

7.1 Engineering Ethics

It is critical to consider the engineering code of ethics policy produced by the American Society of Mechanical Engineers (ASME) during the advancement of the smart wearable device project. These policies ensure that the work of engineers alike ensures and honors the integrity of the engineering profession through practices of human safety, knowledge, impartialness, and innovation. This ensures the betterment of society as a whole where the work of engineers should only aim to improve and assist the population today. In comparison to the smart wearable cast project, it was always considered how we can ensure the safety and replication of the protectiveness of a human arm cast while also expediting the convenience and quality by incorporating the use of sensors to allow it to work autonomously. Using engineering principles based on material science and research, electrical and computer engineering, and computer-aided design, the engineering team of this project was able to use these areas of expertise to create a functional model of the protective smart cast device. It was made sure to abide by all the policies of sustainability of material due to the ability to reuse this device, improve the overall quality of healthcare of patients due to its ability of efficiency in securing the human wrist for fractured arms, and only worked within reason in areas of competence to get each task done appropriately and effectively.

7.2 Societal Impact

The smart arm cast will be beneficial to all patients seeking medical attention for arm-related fractures. This area is significant in the healthcare industry where people of all ages regardless of age, sex, or lifestyle could be affected. According to journal articles from the National Library of Medicine, in the United States alone, it was projected that the national average of fractures was supposed to increase to over 3 million cases by 2025 (Amin et al., 2014). Statistics also show that arm-related fractures occur in 1 out of 100 children and are most common for children ages 5 to 14 years old to account for the highest percentage of arm fractures at a staggering 34% (Rafi and Tiwari, 2023). This project prioritized the safety of users by incorporating sensors to monitor the heart rate and blood-oxygen levels during the healing process and help prevent further movement around the injury by fixation of the wrist. Millions of people have the potential to be treated with a form of casting making this project incredibly important for providing efficient protection with new technology.

7.3 Global Impact

Due to the accessibility and ease of use with the Smart Arm Cast, the effectiveness of this product would not change if used by a patient in a different region of the world . Due to the minimal interference on the patient's body, with its primary function of arm compression during fractures and secondary of recording a patient's heart rate, the cast is usable by patients across the spectrum of race, gender, sex, or culture. Along with this, the cast itself is reusable across patients, allowing healthcare providers in regions of limited access to outside assistance to use the cast across multiple patients over time. This however, can only be accomplished by the maintenance of cast and its equipment, bringing the issue of material availability in other regions of the world into consideration. With the use of the electrical and wiring equipment, damage to just one of the pieces could render the cast unusable without repairing or replacing the part. This could become an issue should a region cannot gain access to the base parts used in this project. Furthermore, the use of electrical equipment and the need to 3D-Print the cast itself could become an issue if a region does not have access to either a 3D Printer or the TPU filament to make the cast model.

7.4 Environmental Impact

During the design process of the cast, a more sustainable approach was taken when considering the kinds of materials that would be used while also having properties such as durability, flexibility, and eco-friendliness. According to the ICP DAS- Biomedical Polymers, the professional provider of medical thermoplastic polyurethane (TPU) stated that TPU is a recyclable material that can be reprocessed, minimizing environmental impacts with less amount of waste generated and that it also reduces the carbon footprint and promotes stability (The Professional Provider of Medical TPU, 2021). Additionally, this project impacts the environment by requiring a significant amount of energy due to the wiring of the sensors, motor, and Arduino which can evidently contribute to carbon emissions, but in a more beneficial way having a well-functioning smart cast, it can limit the number of visits that are taken to a health facility mitigating the risk of carbon emissions and environmental harm.

7.5 Codes and Standards

Considering various codes and standards held by professional and standardization organizations is significant in all engineering initiatives, and proved to be utilized and effective during the development of a smart wearable arm cast. Some of the key references to codes and standards that have been published are the Code of Ethics from the ASME, the International Organization for Standardization (ISO), the American Society of Testing and Materials (ASTM), and the Institute of Electrical and Electronics Engineers (IEEE). Specific standards referenced were general details explained from the *Standard Guide for Evaluation of Thermoplastic Polyurethane Solids and Solutions for Biomedical Applications* published by the ASTM. From this, there is a standard that approves Thermoplastic Polyurethanes to be designed and molded into specific shapes that can be used in biomedical applications. This reference serves as a standard held by the team for the use of the TPU to be used in the method we intend to. A smart wearable cast is an appropriate method of using the material to help the healthcare industry and aid us in the project. Other significant standards that help guide the nature of the smart wearable device project is the *IEEE Standard for Wearable Consumer Electronic Devices--Overview and Architecture*. Standards that proved helpful are “4.1 Medical Devices” and “6.2.4.2 Technical requirements.” Standard 4.1 provides a brief overview on the implementation of consumer electronics in medical devices. One of the devices approved for medical applications is sensors providing biometric data such as blood pressure and heart rate. This provides assurance that the use of sensor technology such as the MAX30102 pulse oximeter combined with the LCD is free to use for biomechanical purposes in our wearable device project. Additionally, standard 6.2.4.2 provides general guidelines for the use of wearable devices for mechanical electrical, electronic, programmable electronics, and software purposes. This guide delves deeper in the protection of users and patients with safety priorities and precautions (IEEE, 2022). Because the project deals with the utilization of smart electronic equipment such as sensors and a mechanical motor, it is important to consider all regulations that involve wearable consumer electronics and how to implement them safely.

7.6 Economic Factors

The economic impact of the smart arm cast goes beyond the realm of healthcare and potentially benefits various stakeholders and sectors of the economy. In the midst of this impact it is important for patient outcomes to improve and for the efficiency in treatment of an arm injury to enhance which leads to cost saving for insurers and the healthcare systems. The materials that this product consists of are two flat pressure sensors that costs $6.06 each, an Arduino costing $23.01, a servo motor that costs $5.95, a pulse oximeter sensor that costs $7, a rechargeable battery power supply costing \$12.99, an circuit push button costing \$0.10, the liquid crystal display costing \$5, additional breadboard wires that cost \$6.98, and the cotton sock that cost \$6.92. By incorporating these products and materials in the smart arm cast it provides unique advantages with access to new technology and data for patients. According to SainSmart, a nationally recognized brand who manufactures 3D printers and filaments, the average cost of TPU filament is \$29.99 for 0.8kg, meaning it has a cost of \$37.48 per kilogram (SainSmart, 2024). The average specific gravity of TPU is 1.33. With this information given, the following equation calculating the cost of TPU filament per cubic inch can be used:

$$\frac{Cost}{in^3} = 16.39\,\frac{cm^3}{in^3} \cdot Specific\ Gravity \cdot \frac{Cost}{Kg} \cdot \frac{1}{1{,}000} \cdot \frac{Kg}{Cm^3}$$

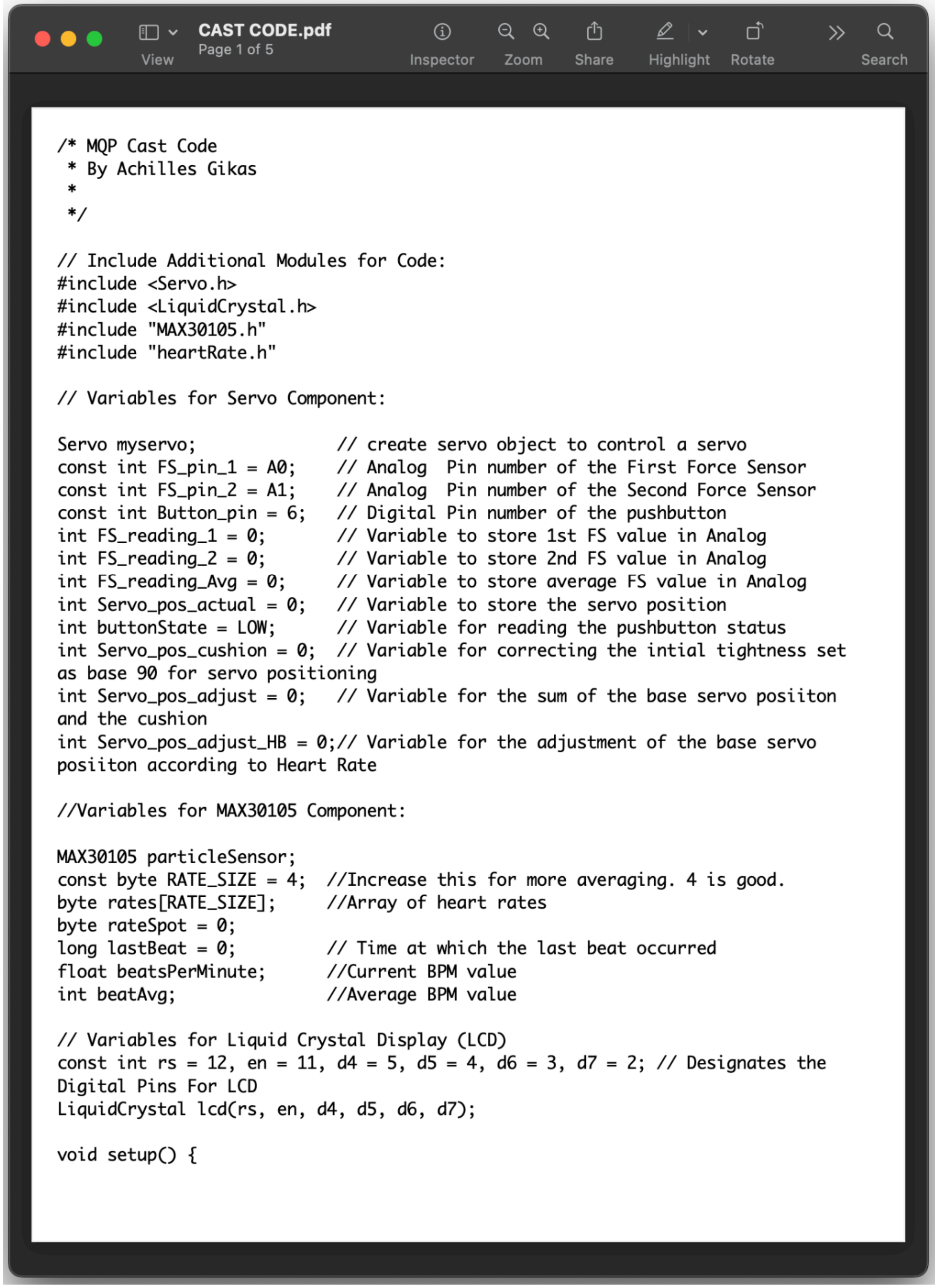
The total cost per cubic inch of TPU filament is \$0.817. The overall cast structure volume measures 20.3235 cubic inches, meaning that to print the short arm cast out of TPU is \$16.66. The totaling cost of equipment and complete assembly of the smart cast is \$96.73. According to a journal article that evaluated research on the cost of materials for immobilization casting on behalf of the Association of Military Surgeons of the United States (ASMUS), they found that the cost of casting to treat short arm fractures with plaster of paris was \$12.90 and \$15.90 when using fiberglass (Kowalski et al., 2002). Although the average price for treating arm injuries is currently less than the proposed smart cast model, the overall quality and efficiency of current casting methods lack value due to the incorporation of new technology that provides biometric data to the patient and autonomous fixation.

8.0 Conclusion

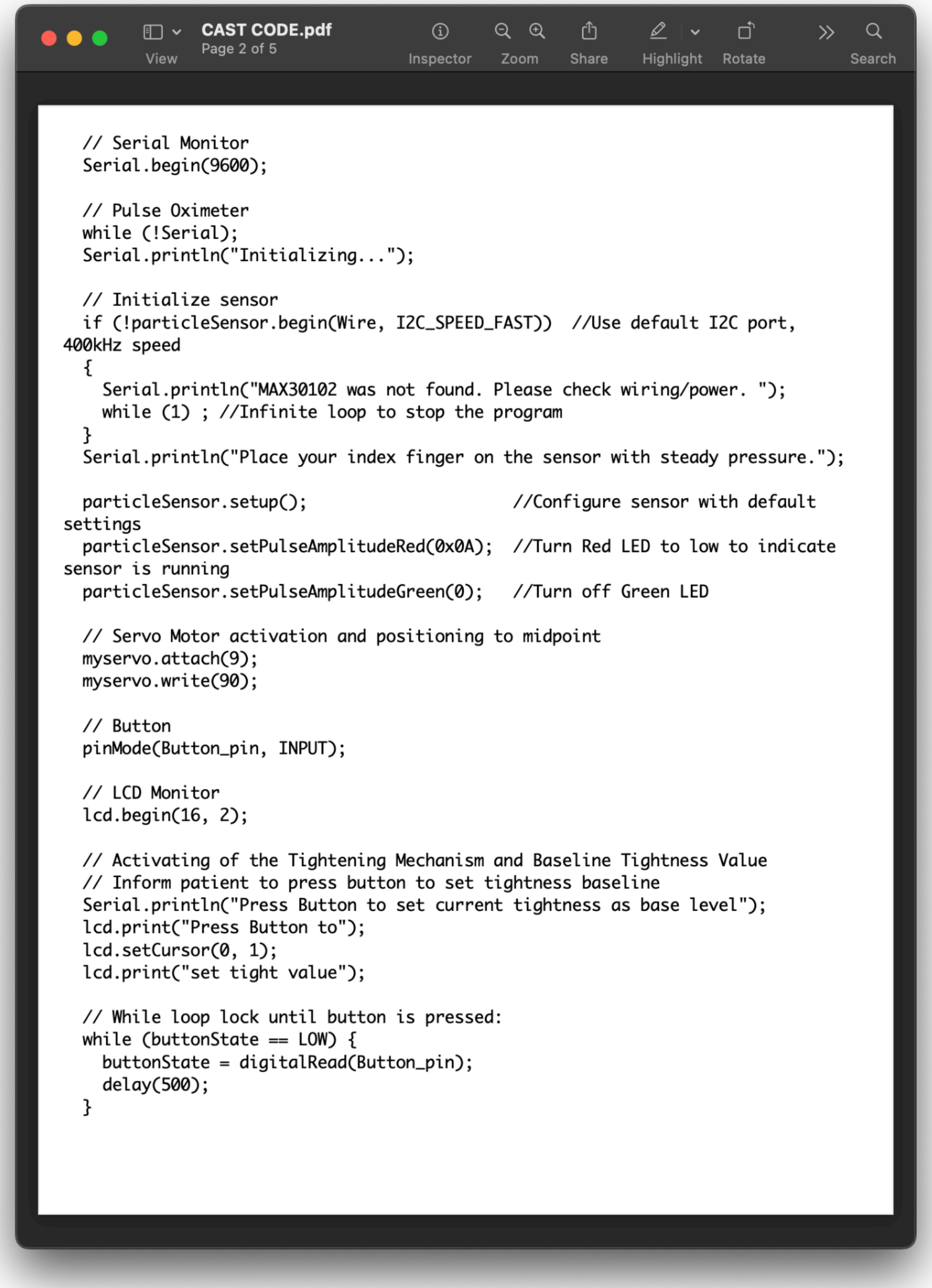
The servo and sensor enhanced smart arm cast delves into a new world of opportunities within the healthcare industry by supplying injured patients with an alternative casting method that incorporates technology and differing materials not currently being used for optimal and autonomous immobilization. The effects of this device are endless with an average of millions of arm-related fractures happening year round in the United States alone. With the use of equipment such as strain gauge pressure sensors, a pulse oximeter sensor, an Arduino Nano R3, a liquid crystal display, a servo motor, nylon fisher wire, a PLA fabricated pulley wheel, battery power source, a miniature breadboard, and additional circuit wires, a functional cast made of Thermoplastic Polyurethane (TPU), a unique material with qualities making it durable for protection and flexible to tighten and loosen, can be assembled and function to fixate an injured patient's wrist on its own while supplying the user's heart rate. One generated device was created and assembled for initial testing that proved possible performance and accurate pressure readings within the smart cast. Baseline testing consisted of code and sensor calibration, max tension and torque readings, and short term performance of the entirely assembled cast. These tests provided data that gave insight on optimal design of the cast body and tightening mechanism, as well as casting light on future work and recommendation for the continuation of this project. This project considered the basic needs for an alternative short arm cast and proved the possibility of creating a smart device accessible to injured patients that could functionally tighten and loosen based on pressure and heart rate.

Appendix A: Arduino Code Used in Project

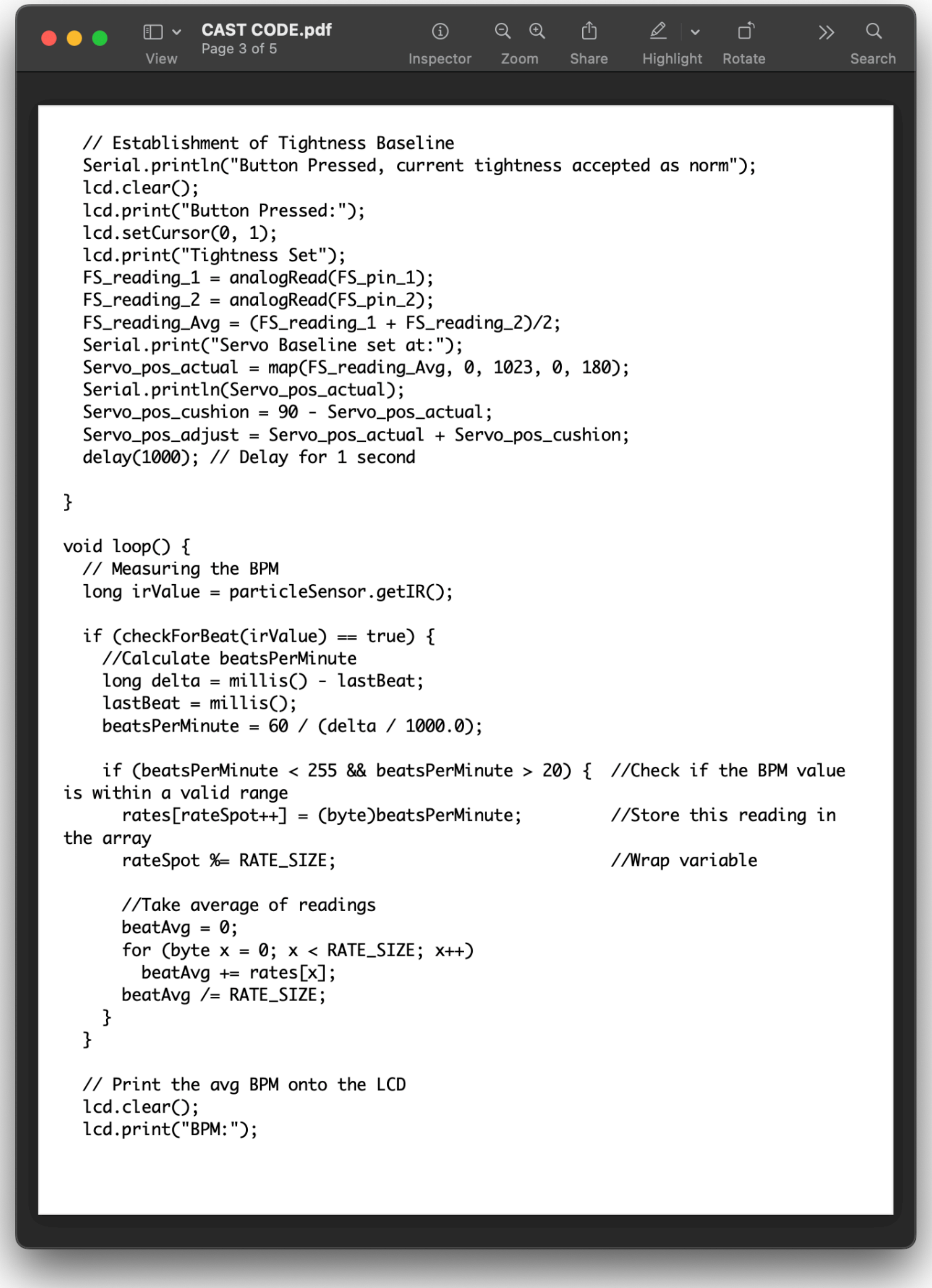
Primary Casting Code:



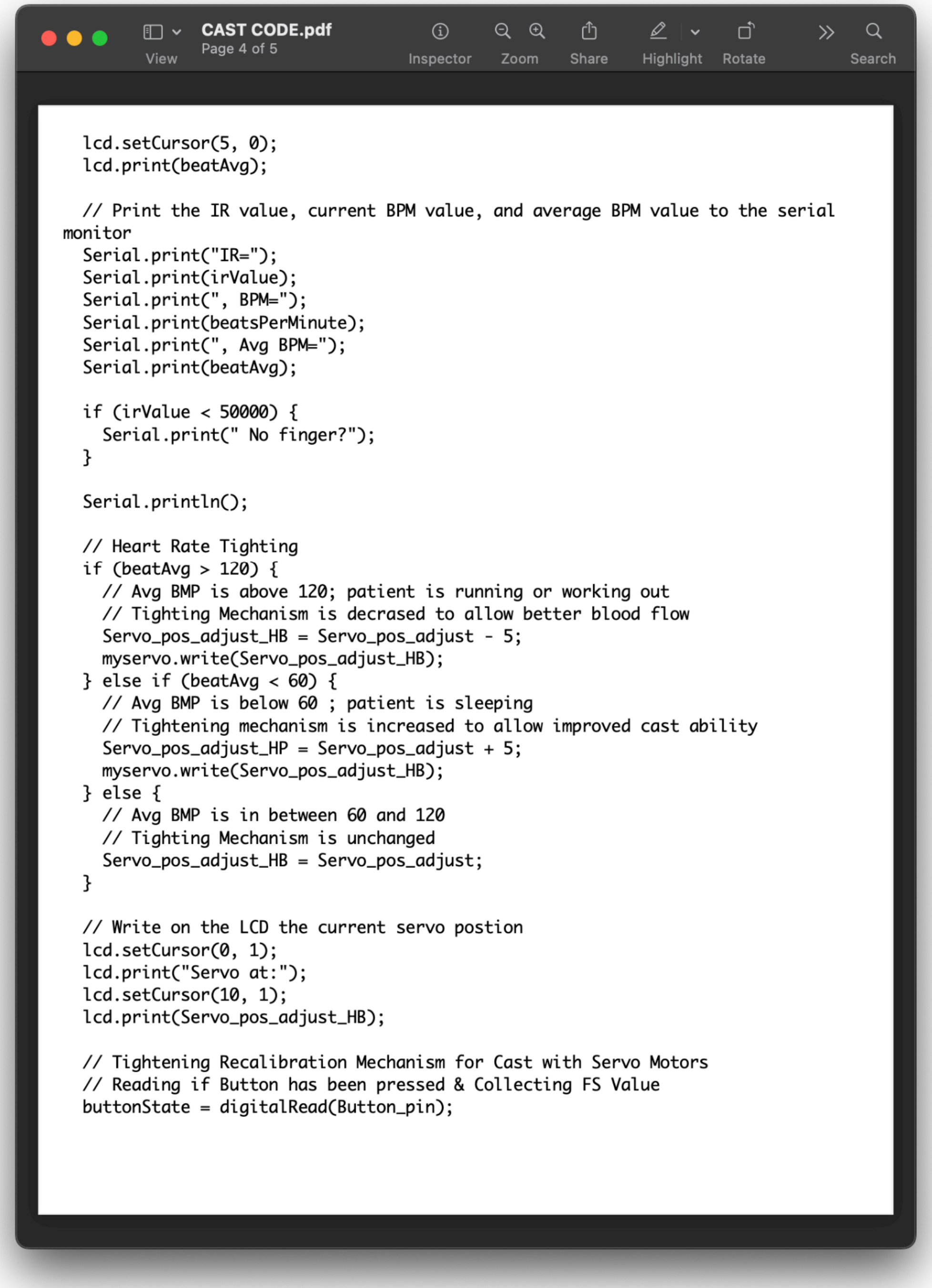
```
/* MQP Cast Code
 * By Achilles Gikas
 *
 */

// Include Additional Modules for Code:
#include <Servo.h>
#include <LiquidCrystal.h>
#include "MAX30105.h"
#include "heartRate.h"

// Variables for Servo Component:

Servo myservo;              // create servo object to control a servo
const int FS_pin_1 = A0;    // Analog  Pin number of the First Force Sensor
const int FS_pin_2 = A1;    // Analog  Pin number of the Second Force Sensor
const int Button_pin = 6;   // Digital Pin number of the pushbutton
int FS_reading_1 = 0;       // Variable to store 1st FS value in Analog
int FS_reading_2 = 0;       // Variable to store 2nd FS value in Analog
int FS_reading_Avg = 0;     // Variable to store average FS value in Analog
int Servo_pos_actual = 0;   // Variable to store the servo position
int buttonState = LOW;      // Variable for reading the pushbutton status
int Servo_pos_cushion = 0;  // Variable for correcting the intial tightness set
as base 90 for servo positioning
int Servo_pos_adjust = 0;   // Variable for the sum of the base servo posiiton
and the cushion
int Servo_pos_adjust_HB = 0;// Variable for the adjustment of the base servo
posiiton according to Heart Rate

//Variables for MAX30105 Component:

MAX30105 particleSensor;
const byte RATE_SIZE = 4;  //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE];     //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0;         // Time at which the last beat occurred
float beatsPerMinute;      //Current BPM value
int beatAvg;               //Average BPM value

// Variables for Liquid Crystal Display (LCD)
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2; // Designates the
Digital Pins For LCD
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
```



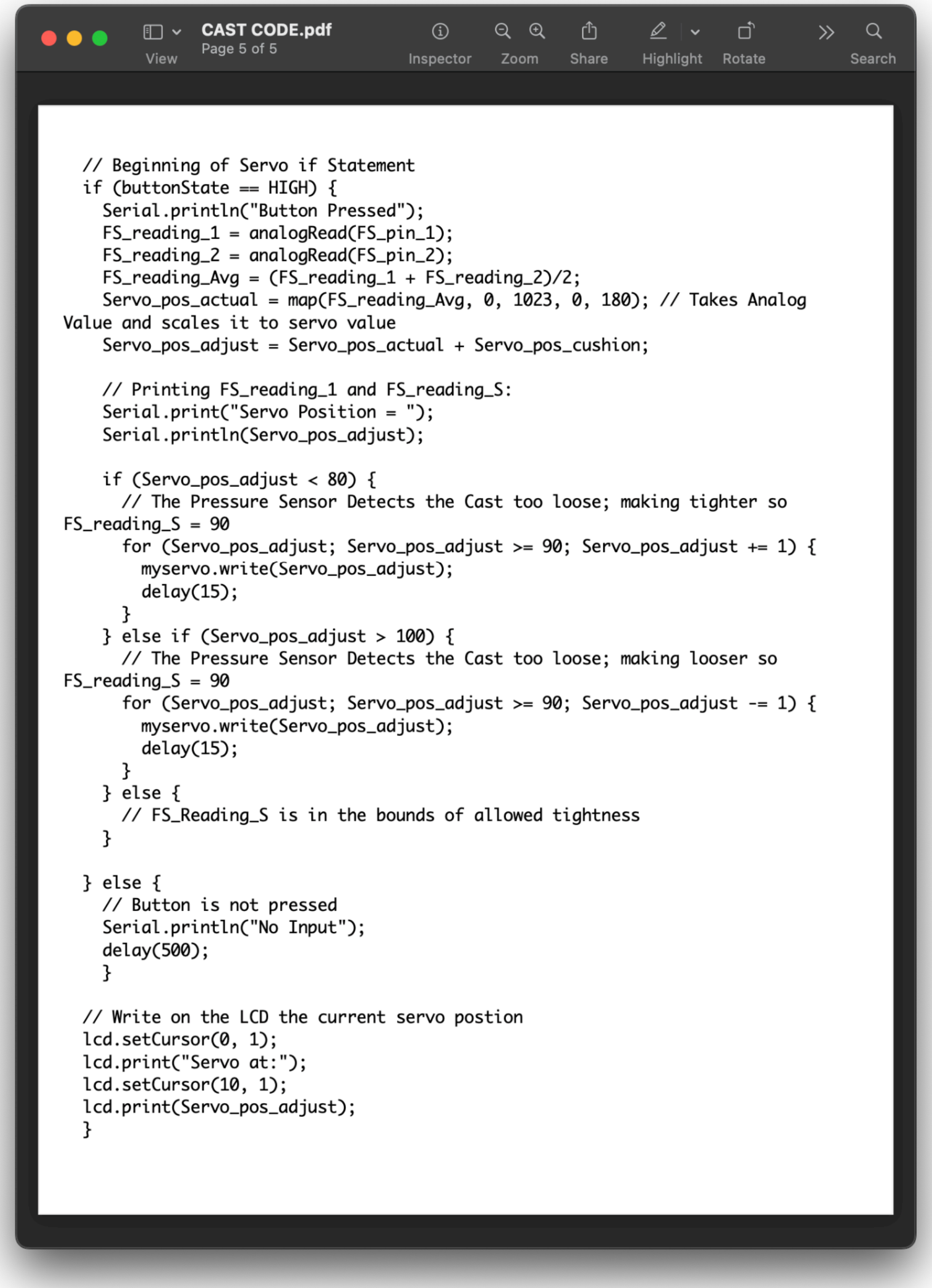
```
  // Serial Monitor
  Serial.begin(9600);

  // Pulse Oximeter
  while (!Serial);
  Serial.println("Initializing...");

  // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST))  //Use default I2C port,
400kHz speed
  {
    Serial.println("MAX30102 was not found. Please check wiring/power. ");
    while (1) ; //Infinite loop to stop the program
  }
  Serial.println("Place your index finger on the sensor with steady pressure.");

  particleSensor.setup();                       //Configure sensor with default
settings
  particleSensor.setPulseAmplitudeRed(0x0A);  //Turn Red LED to low to indicate
sensor is running
  particleSensor.setPulseAmplitudeGreen(0);   //Turn off Green LED

  // Servo Motor activation and positioning to midpoint
  myservo.attach(9);
  myservo.write(90);

  // Button
  pinMode(Button_pin, INPUT);

  // LCD Monitor
  lcd.begin(16, 2);

  // Activating of the Tightening Mechanism and Baseline Tightness Value
  // Inform patient to press button to set tightness baseline
  Serial.println("Press Button to set current tightness as base level");
  lcd.print("Press Button to");
  lcd.setCursor(0, 1);
  lcd.print("set tight value");

  // While loop lock until button is pressed:
  while (buttonState == LOW) {
    buttonState = digitalRead(Button_pin);
    delay(500);
  }
```



```
  // Establishment of Tightness Baseline
  Serial.println("Button Pressed, current tightness accepted as norm");
  lcd.clear();
  lcd.print("Button Pressed:");
  lcd.setCursor(0, 1);
  lcd.print("Tightness Set");
  FS_reading_1 = analogRead(FS_pin_1);
  FS_reading_2 = analogRead(FS_pin_2);
  FS_reading_Avg = (FS_reading_1 + FS_reading_2)/2;
  Serial.print("Servo Baseline set at:");
  Servo_pos_actual = map(FS_reading_Avg, 0, 1023, 0, 180);
  Serial.println(Servo_pos_actual);
  Servo_pos_cushion = 90 - Servo_pos_actual;
  Servo_pos_adjust = Servo_pos_actual + Servo_pos_cushion;
  delay(1000); // Delay for 1 second

}

void loop() {
  // Measuring the BPM
  long irValue = particleSensor.getIR();

  if (checkForBeat(irValue) == true) {
    //Calculate beatsPerMinute
    long delta = millis() - lastBeat;
    lastBeat = millis();
    beatsPerMinute = 60 / (delta / 1000.0);

    if (beatsPerMinute < 255 && beatsPerMinute > 20) {  //Check if the BPM value
is within a valid range
      rates[rateSpot++] = (byte)beatsPerMinute;         //Store this reading in
the array
      rateSpot %= RATE_SIZE;                            //Wrap variable

      //Take average of readings
      beatAvg = 0;
      for (byte x = 0; x < RATE_SIZE; x++)
        beatAvg += rates[x];
      beatAvg /= RATE_SIZE;
    }
  }

  // Print the avg BPM onto the LCD
  lcd.clear();
  lcd.print("BPM:");
```

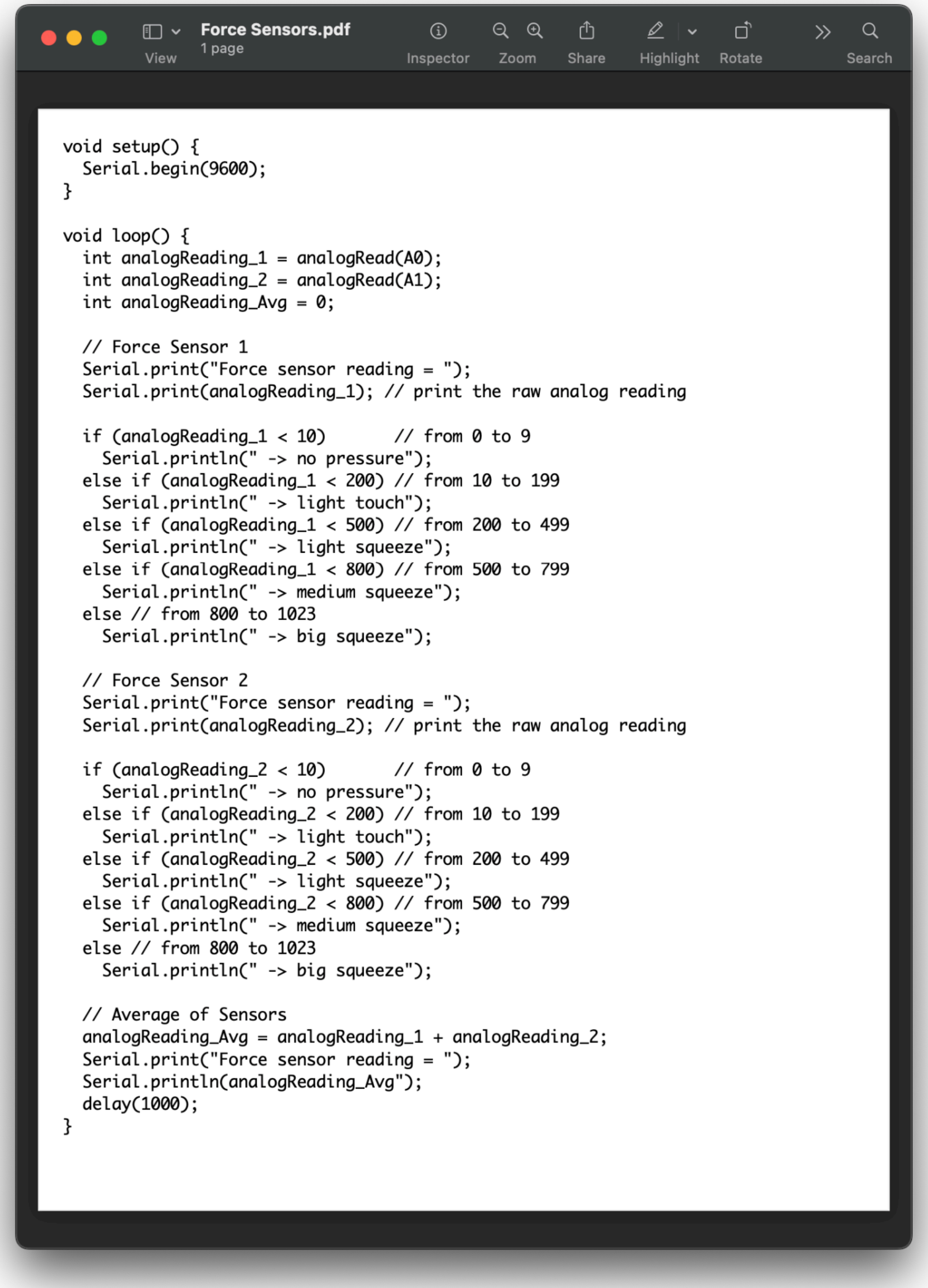


```
  lcd.setCursor(5, 0);
  lcd.print(beatAvg);

  // Print the IR value, current BPM value, and average BPM value to the serial
monitor
  Serial.print("IR=");
  Serial.print(irValue);
  Serial.print(", BPM=");
  Serial.print(beatsPerMinute);
  Serial.print(", Avg BPM=");
  Serial.print(beatAvg);

  if (irValue < 50000) {
    Serial.print(" No finger?");
  }

  Serial.println();

  // Heart Rate Tighting
  if (beatAvg > 120) {
    // Avg BMP is above 120; patient is running or working out
    // Tighting Mechanism is decrased to allow better blood flow
    Servo_pos_adjust_HB = Servo_pos_adjust - 5;
    myservo.write(Servo_pos_adjust_HB);
  } else if (beatAvg < 60) {
    // Avg BMP is below 60 ; patient is sleeping
    // Tightening mechanism is increased to allow improved cast ability
    Servo_pos_adjust_HP = Servo_pos_adjust + 5;
    myservo.write(Servo_pos_adjust_HB);
  } else {
    // Avg BMP is in between 60 and 120
    // Tighting Mechanism is unchanged
    Servo_pos_adjust_HB = Servo_pos_adjust;
  }

  // Write on the LCD the current servo postion
  lcd.setCursor(0, 1);
  lcd.print("Servo at:");
  lcd.setCursor(10, 1);
  lcd.print(Servo_pos_adjust_HB);

  // Tightening Recalibration Mechanism for Cast with Servo Motors
  // Reading if Button has been pressed & Collecting FS Value
  buttonState = digitalRead(Button_pin);
```



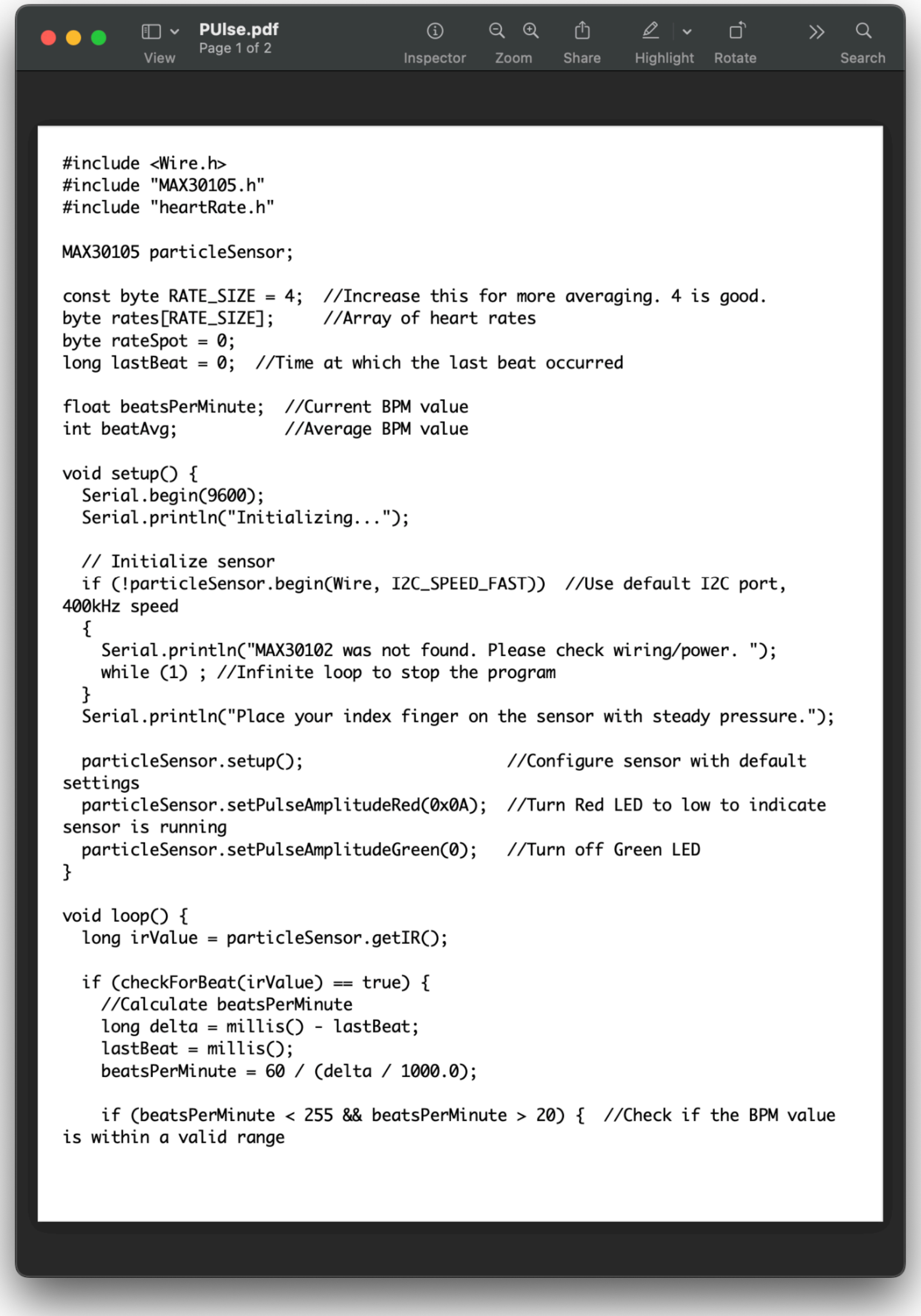
```
  // Beginning of Servo if Statement
  if (buttonState == HIGH) {
    Serial.println("Button Pressed");
    FS_reading_1 = analogRead(FS_pin_1);
    FS_reading_2 = analogRead(FS_pin_2);
    FS_reading_Avg = (FS_reading_1 + FS_reading_2)/2;
    Servo_pos_actual = map(FS_reading_Avg, 0, 1023, 0, 180); // Takes Analog
Value and scales it to servo value
    Servo_pos_adjust = Servo_pos_actual + Servo_pos_cushion;

    // Printing FS_reading_1 and FS_reading_S:
    Serial.print("Servo Position = ");
    Serial.println(Servo_pos_adjust);

    if (Servo_pos_adjust < 80) {
      // The Pressure Sensor Detects the Cast too loose; making tighter so
FS_reading_S = 90
      for (Servo_pos_adjust; Servo_pos_adjust >= 90; Servo_pos_adjust += 1) {
        myservo.write(Servo_pos_adjust);
        delay(15);
      }
    } else if (Servo_pos_adjust > 100) {
      // The Pressure Sensor Detects the Cast too loose; making looser so
FS_reading_S = 90
      for (Servo_pos_adjust; Servo_pos_adjust >= 90; Servo_pos_adjust -= 1) {
        myservo.write(Servo_pos_adjust);
        delay(15);
      }
    } else {
      // FS_Reading_S is in the bounds of allowed tightness
    }

  } else {
    // Button is not pressed
    Serial.println("No Input");
    delay(500);
    }

  // Write on the LCD the current servo postion
  lcd.setCursor(0, 1);
  lcd.print("Servo at:");
  lcd.setCursor(10, 1);
  lcd.print(Servo_pos_adjust);
  }
```

Strain Gauge Calibration & Testing

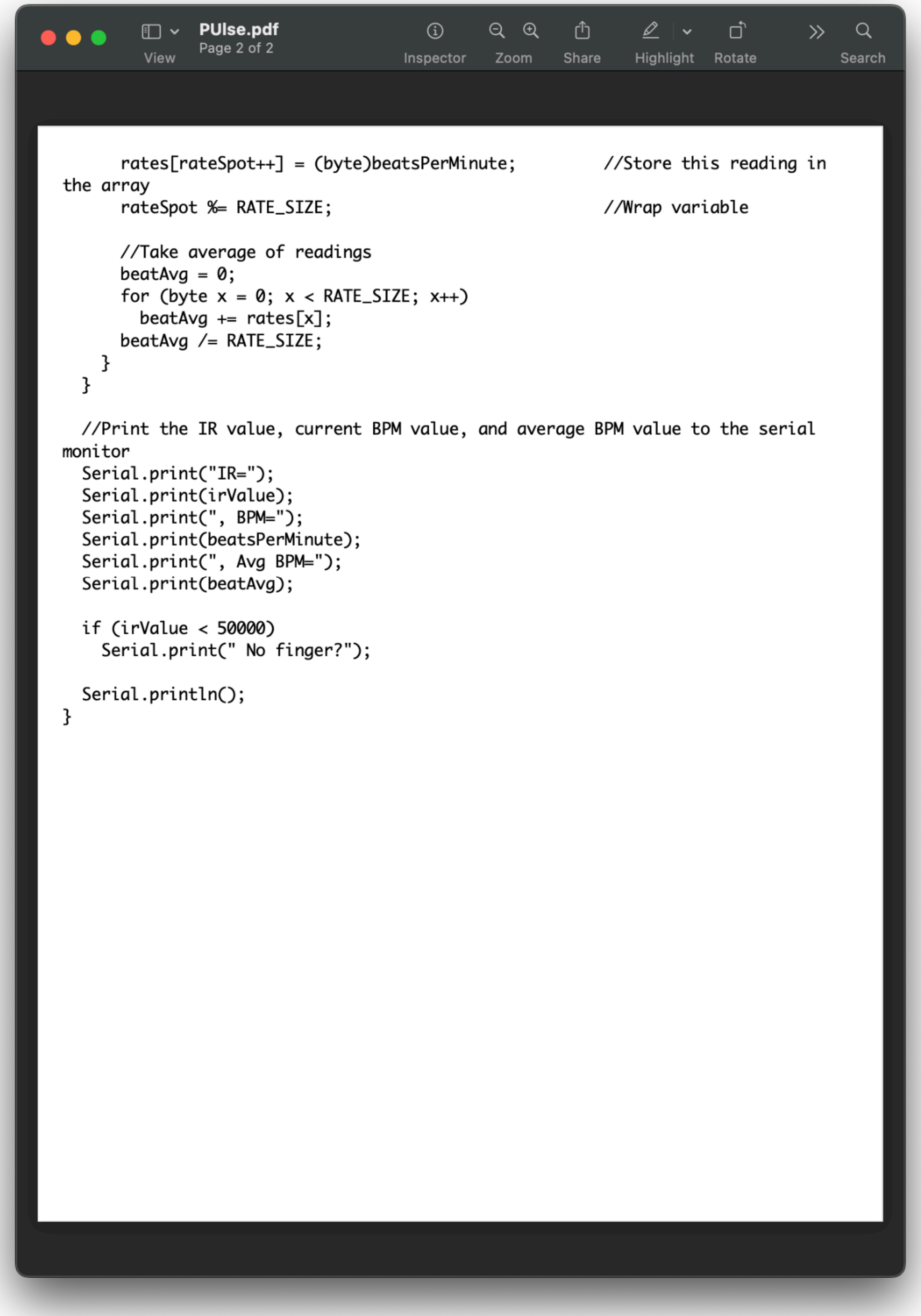


```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int analogReading_1 = analogRead(A0);
  int analogReading_2 = analogRead(A1);
  int analogReading_Avg = 0;

  // Force Sensor 1
  Serial.print("Force sensor reading = ");
  Serial.print(analogReading_1); // print the raw analog reading

  if (analogReading_1 < 10)       // from 0 to 9
    Serial.println(" -> no pressure");
  else if (analogReading_1 < 200) // from 10 to 199
    Serial.println(" -> light touch");
  else if (analogReading_1 < 500) // from 200 to 499
    Serial.println(" -> light squeeze");
  else if (analogReading_1 < 800) // from 500 to 799
    Serial.println(" -> medium squeeze");
  else // from 800 to 1023
    Serial.println(" -> big squeeze");

  // Force Sensor 2
  Serial.print("Force sensor reading = ");
  Serial.print(analogReading_2); // print the raw analog reading

  if (analogReading_2 < 10)       // from 0 to 9
    Serial.println(" -> no pressure");
  else if (analogReading_2 < 200) // from 10 to 199
    Serial.println(" -> light touch");
  else if (analogReading_2 < 500) // from 200 to 499
    Serial.println(" -> light squeeze");
  else if (analogReading_2 < 800) // from 500 to 799
    Serial.println(" -> medium squeeze");
  else // from 800 to 1023
    Serial.println(" -> big squeeze");

  // Average of Sensors
  analogReading_Avg = analogReading_1 + analogReading_2;
  Serial.print("Force sensor reading = ");
  Serial.println(analogReading_Avg");
  delay(1000);
}
```

Pulse Oximeter Calibration & Testing

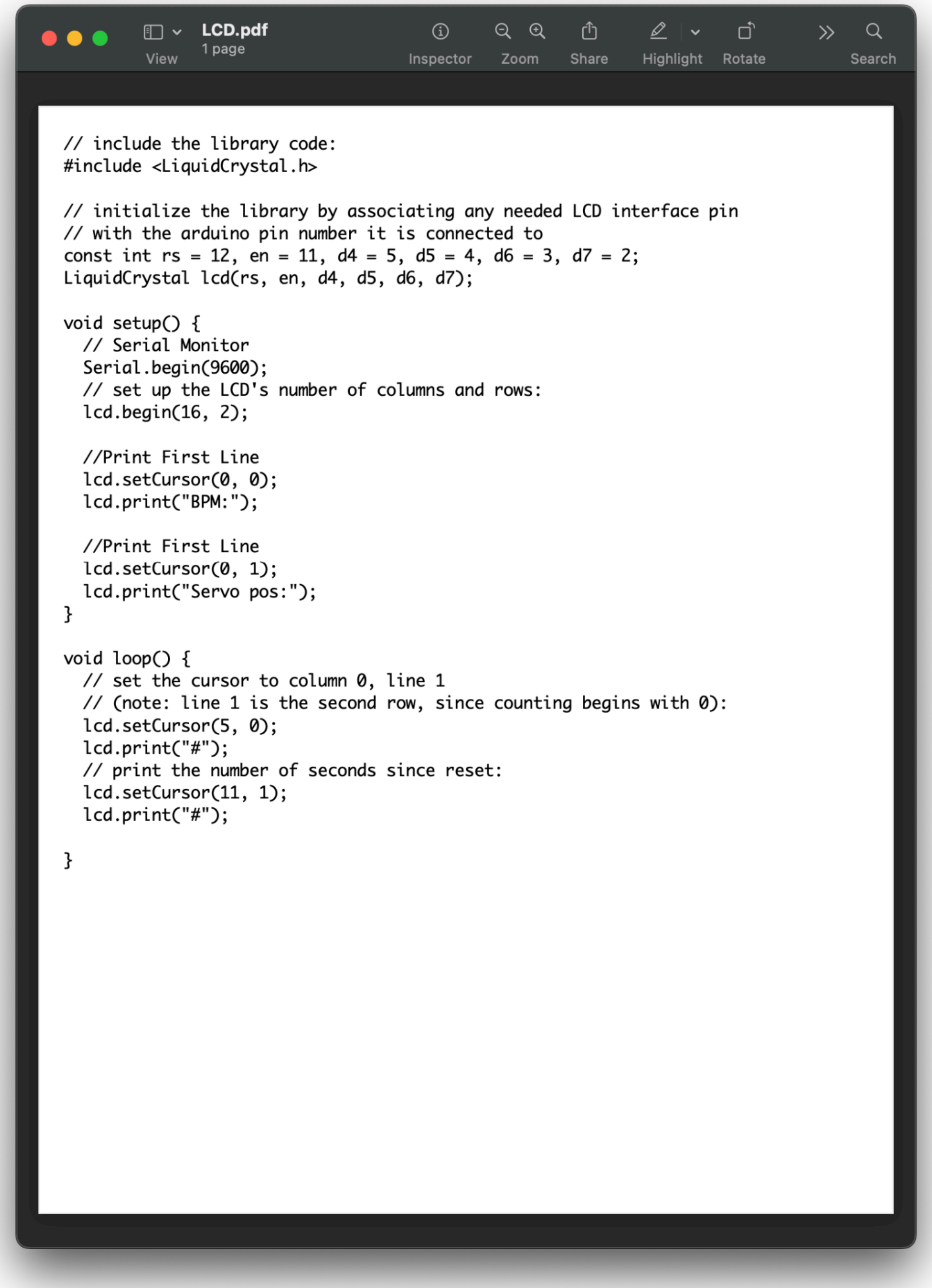


```
#include <Wire.h>
#include "MAX30105.h"
#include "heartRate.h"

MAX30105 particleSensor;

const byte RATE_SIZE = 4;  //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE];     //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0;  //Time at which the last beat occurred

float beatsPerMinute;  //Current BPM value
int beatAvg;           //Average BPM value

void setup() {
  Serial.begin(9600);
  Serial.println("Initializing...");

  // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST))  //Use default I2C port,
400kHz speed
  {
    Serial.println("MAX30102 was not found. Please check wiring/power. ");
    while (1) ; //Infinite loop to stop the program
  }
  Serial.println("Place your index finger on the sensor with steady pressure.");

  particleSensor.setup();                      //Configure sensor with default
settings
  particleSensor.setPulseAmplitudeRed(0x0A);  //Turn Red LED to low to indicate
sensor is running
  particleSensor.setPulseAmplitudeGreen(0);   //Turn off Green LED
}

void loop() {
  long irValue = particleSensor.getIR();

  if (checkForBeat(irValue) == true) {
    //Calculate beatsPerMinute
    long delta = millis() - lastBeat;
    lastBeat = millis();
    beatsPerMinute = 60 / (delta / 1000.0);

    if (beatsPerMinute < 255 && beatsPerMinute > 20) {  //Check if the BPM value
is within a valid range
```



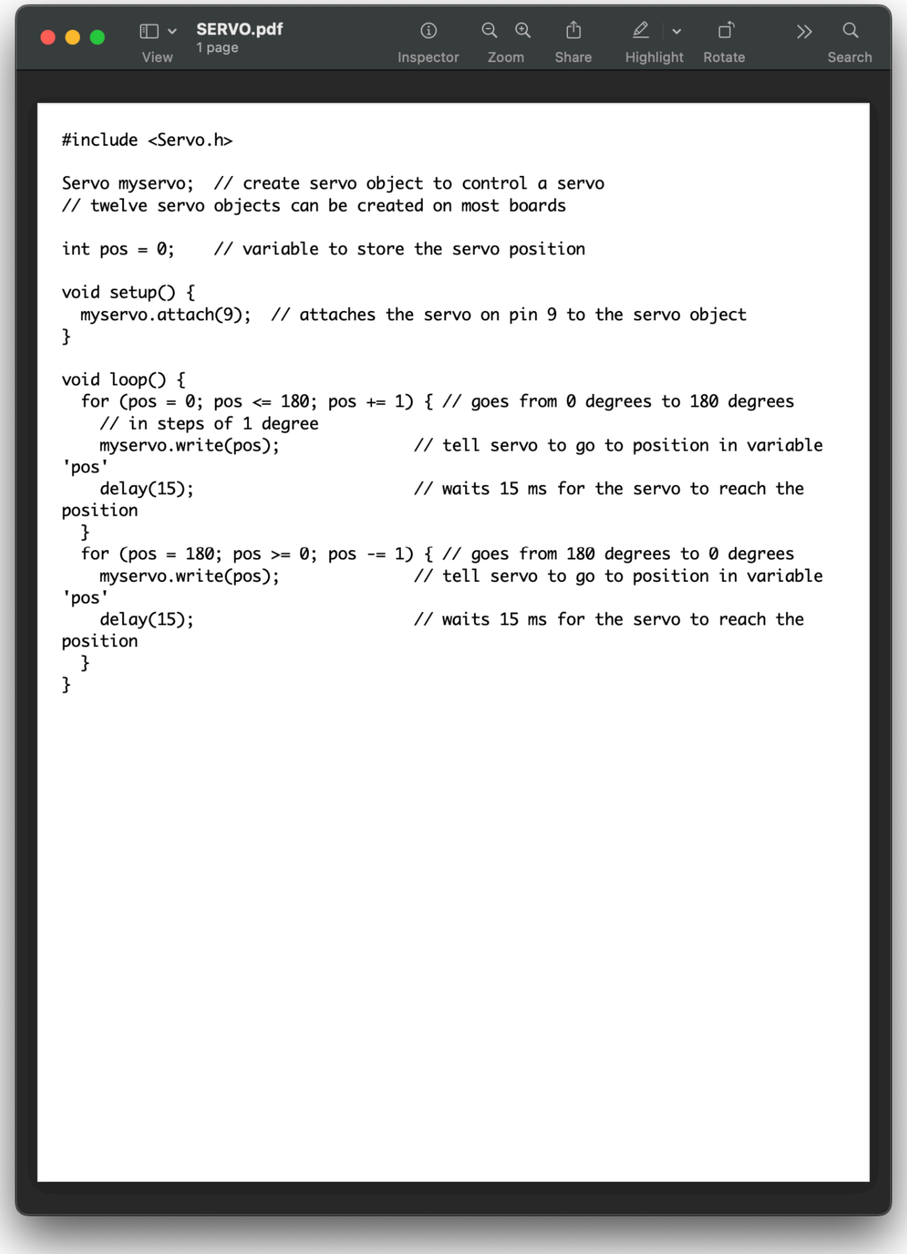
```
      rates[rateSpot++] = (byte)beatsPerMinute;         //Store this reading in
the array
      rateSpot %= RATE_SIZE;                            //Wrap variable

      //Take average of readings
      beatAvg = 0;
      for (byte x = 0; x < RATE_SIZE; x++)
        beatAvg += rates[x];
      beatAvg /= RATE_SIZE;
    }
  }

  //Print the IR value, current BPM value, and average BPM value to the serial
monitor
  Serial.print("IR=");
  Serial.print(irValue);
  Serial.print(", BPM=");
  Serial.print(beatsPerMinute);
  Serial.print(", Avg BPM=");
  Serial.print(beatAvg);

  if (irValue < 50000)
    Serial.print(" No finger?");

  Serial.println();
}
```

LCD Testing



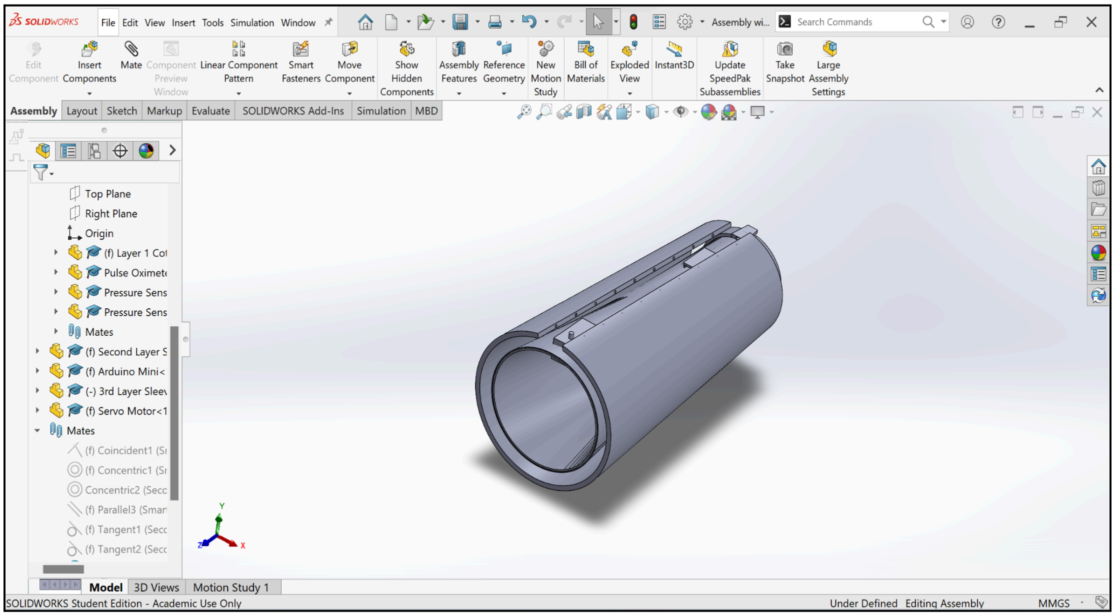
```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  // Serial Monitor
  Serial.begin(9600);
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  //Print First Line
  lcd.setCursor(0, 0);
  lcd.print("BPM:");

  //Print First Line
  lcd.setCursor(0, 1);
  lcd.print("Servo pos:");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(5, 0);
  lcd.print("#");
  // print the number of seconds since reset:
  lcd.setCursor(11, 1);
  lcd.print("#");

}
```

Servo Motor Range Sweep Testing

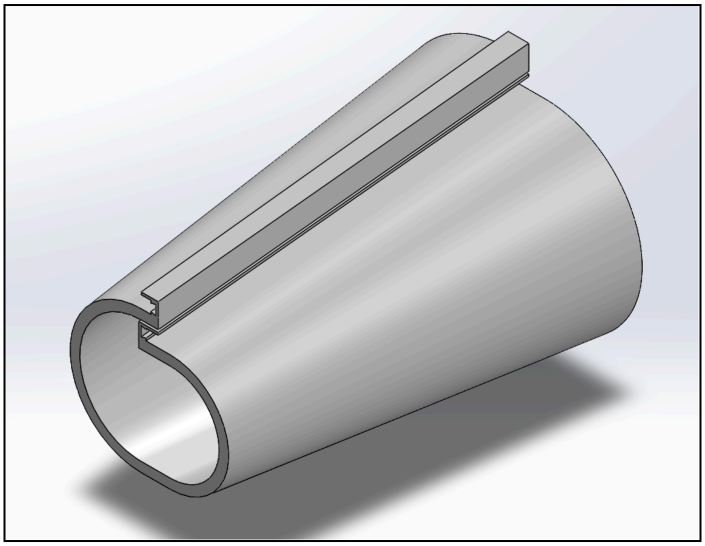


```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo
// twelve servo objects can be created on most boards

int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable
'pos'
    delay(15);                       // waits 15 ms for the servo to reach the
position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable
'pos'
    delay(15);                       // waits 15 ms for the servo to reach the
position
  }
}
```

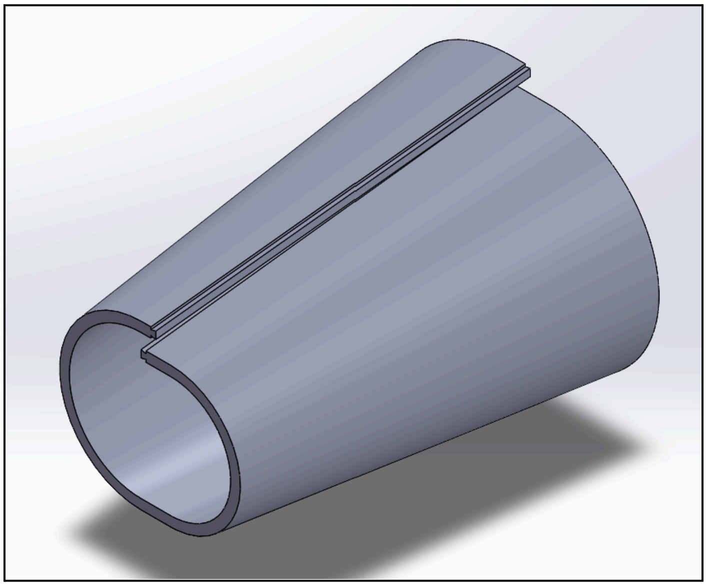
Appendix B: CAD Iterations of Cast Main Body



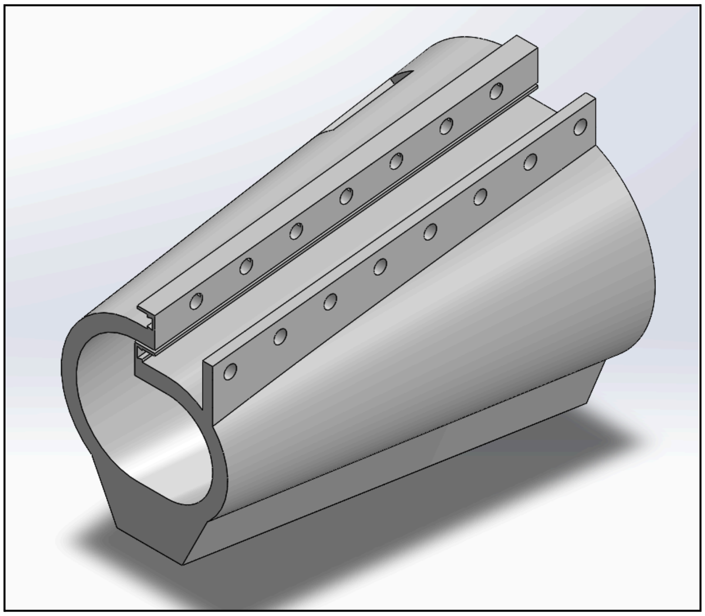
Original SolidWorks Assembly Model of Smart Cast with Three Layers



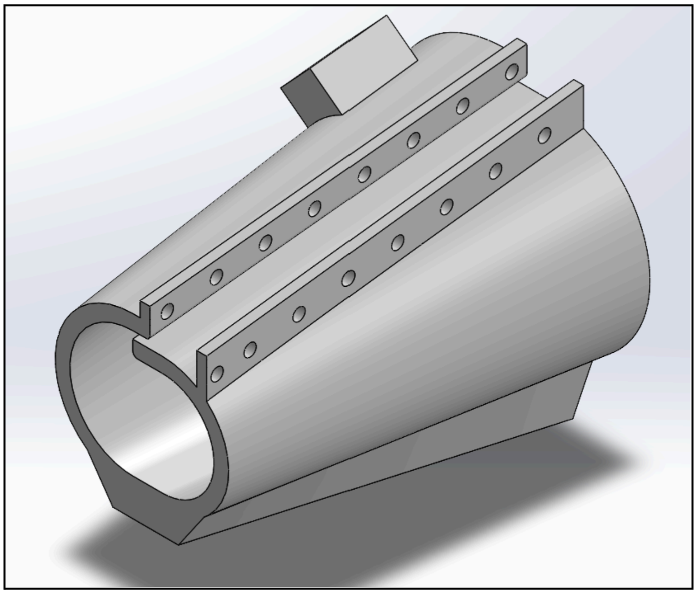
2nd Layer Iteration with Dovetail Slots for 3rd Layer Outer Shell



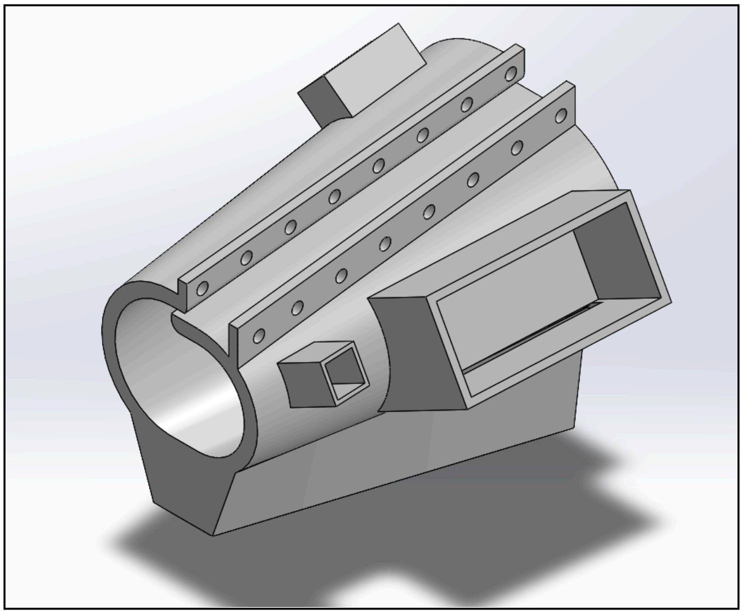
Original 3rd Layer Outer Shell with Dovetail-Shaped Connections



Second Layer Iteration with Dovetail Slots, Tightening Walls, Bottom Loft, and Motor Slot



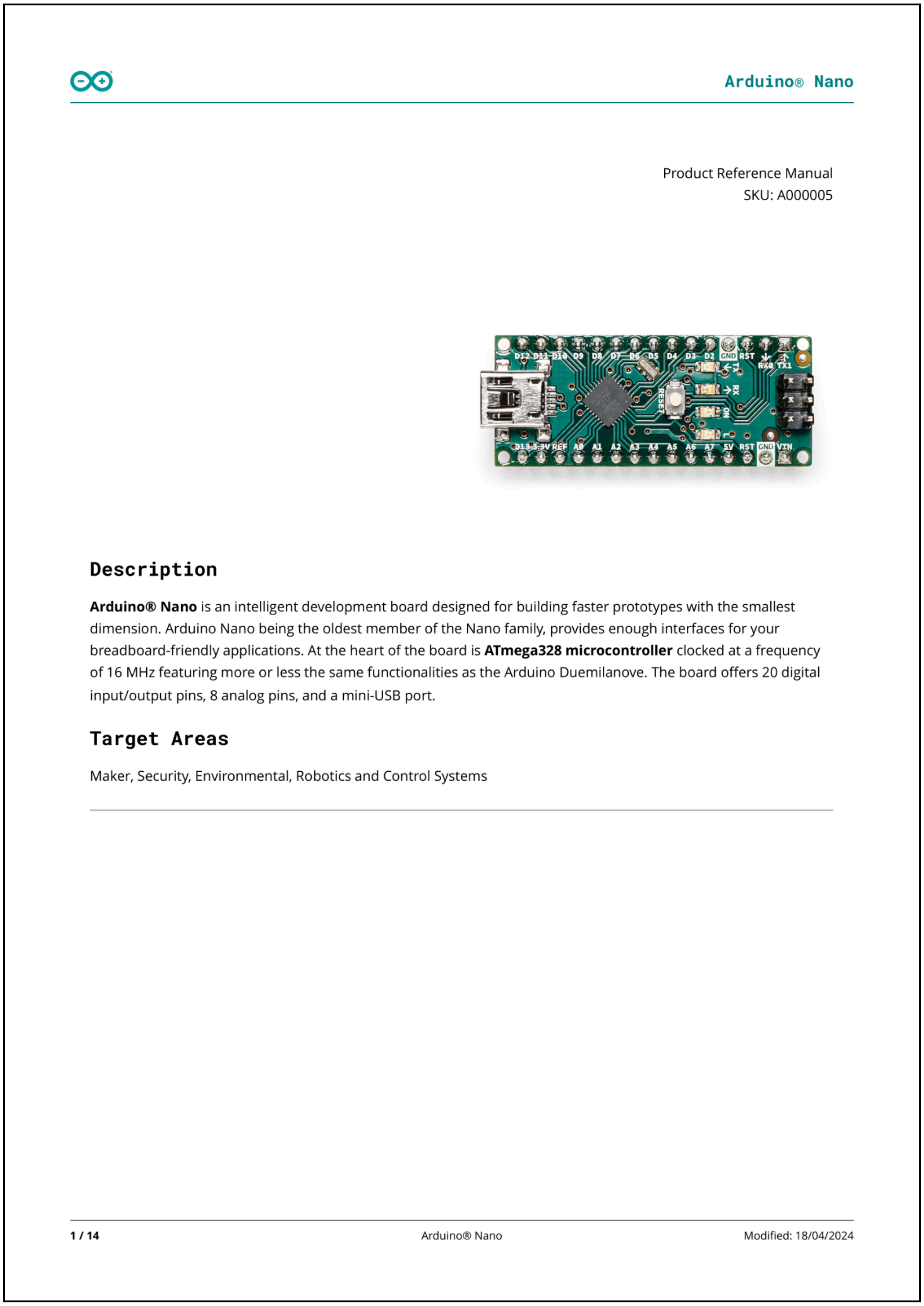
Cast Body Iteration with Removed Dovetails Slots and Updated Motor Case



Cast Body Iteration with Updated Bottom Loft and Added Casing for LCD and Button

Appendix C: Arduino Equipment Spec & Data Sheets

Arduino Nano



Arduino® Nano

Product Reference Manual
SKU: A000005

Description

Arduino® Nano is an intelligent development board designed for building faster prototypes with the smallest dimension. Arduino Nano being the oldest member of the Nano family, provides enough interfaces for your breadboard-friendly applications. At the heart of the board is **ATmega328 microcontroller** clocked at a frequency of 16 MHz featuring more or less the same functionalities as the Arduino Duemilanove. The board offers 20 digital input/output pins, 8 analog pins, and a mini-USB port.

Target Areas

Maker, Security, Environmental, Robotics and Control Systems

1 / 14 Arduino® Nano Modified: 18/04/2024

Features

- **ATmega328** Microcontroller
    - High-performance low-power 8-bit processor
    - Achieve up to 16 MIPS for 16 MHz clock frequency
    - 32 kB of which 2 KB used by bootloader
    - 2 kB internal SRAM
    - 1 kB EEPROM
    - 32 x 8 General Purpose Working Registers
    - Real Time Counter with Separate Oscillator
    - Six PWM Channels
    - Programmable Serial USART
    - Master/Slave SPI Serial Interface
- **Power**
    - Mini-B USB connection
    - 7-15V unregulated external power supply (pin 30)
    - 5V regulated external power supply (pin 27)
- **Sleep Modes**
    - Idle
    - ADC Noise Reduction
    - Power-save
    - Power-down
    - Standby
    - Extended Standby
- **I/O**
    - 20 Digital
    - 8 Analog
    - 6 PWM Output

Contents

1 The Board

1.1 Application Examples

Arduino Nano is the first embedded microcontroller in the Nano series with minimum functionalities, designed for mini projects from the maker community. With a large number of input/output pins gives the advantage of utilizing several serial communications like UART, SPI and I2C. The hardware is compatible with Arduino IDE, Arduino CLI and web editor.

Security: The high-performance and low-power capabilities gives the chance to develop security based applications like access control systems using fingerprint sensors. The flexibility to interface sensors and external devices using serial communication has improved the scope of utility.

Environmental: The low-power feature of the microcontroller and the power supply options for the board has enhanced the ability to implement remote IoT projects related to environmental issues.

Robotics: Robotics has always been the favorite area of exploration for the Maker community and with this tiny embedded hardware you can now create complex and advanced robotic applications.

1.2 Accessories

1.3 Related Products

- Arduino Nano 33 BLE
- Arduino 33 IoT
- Arduino Micro

2 Ratings

2.1 Recommended Operating Conditions

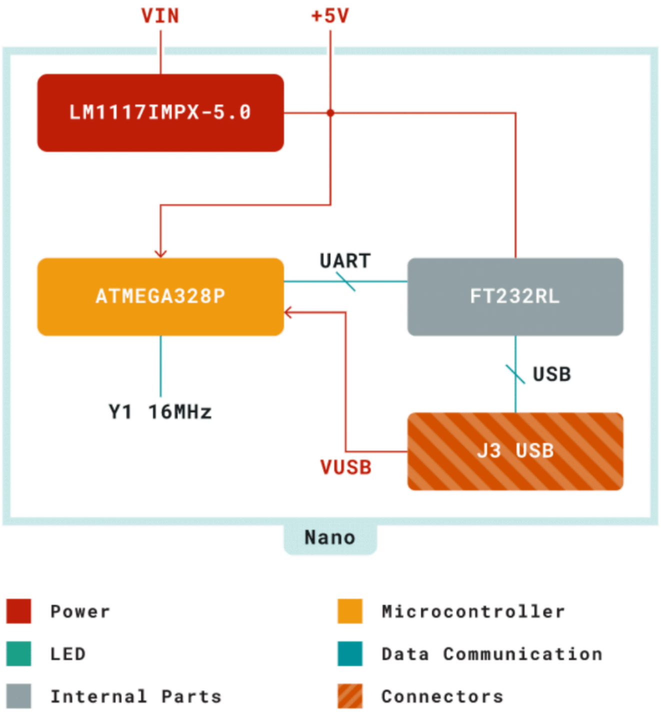
Symbol	Description	Min	Max
	Conservative thermal limits for the whole board:	-40 °C	85 °C

2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
USB VCC	Input supply from USB		TBC		mW
VIN	Input from VIN pad		TBC		mW

3 Functional Overview

3.1 Block Diagram

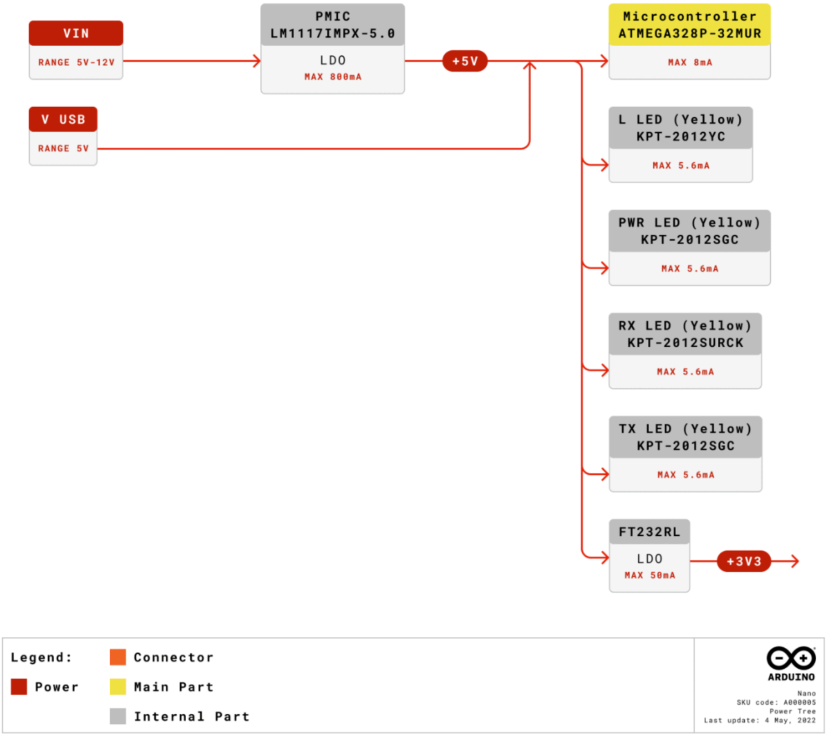


Block Diagram of Arduino Nano

3.2 Processor

The primary processor in the Arduino Nano v3.3 board is the high-performance and low-power 8-bit ATmega328 microcontroller that runs at a clock frequency of 16 MHz. The ability to interface external devices through serial communication supported by the chip with UART TTL (5V), I2C (TWI) and SPI. Arduino Nano can be programmed with Arduino software reducing the entry barriers for new users. Smallest dimension embedded hardware makes it a perfect choice for breadboard-friendly projects from the maker community.

3.3 Power Tree



Power Tree of Arduino Nano

The Arduino Nano can be powered by either the USB port or alternatively via VIN. The input supply of VIN is regulated by an LDO so the supply is limited to 5V for the optimal functioning of the board. There is also another regulator which limits the voltage to 3.3V for powrering the components with low voltage requirements.

4 Board Operation

4.1 Getting Started - IDE

If you want to program your Arduino® Nano while offline you need to install the Arduino® Desktop IDE **[1]** To connect the Arduino Uno to your computer, you'll need a Micro-B USB cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Web Editor

All Arduino® boards, including this one, work out-of-the-box on the Arduino Web Editor **[2]**, by just installing a simple plugin. The Arduino Web Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

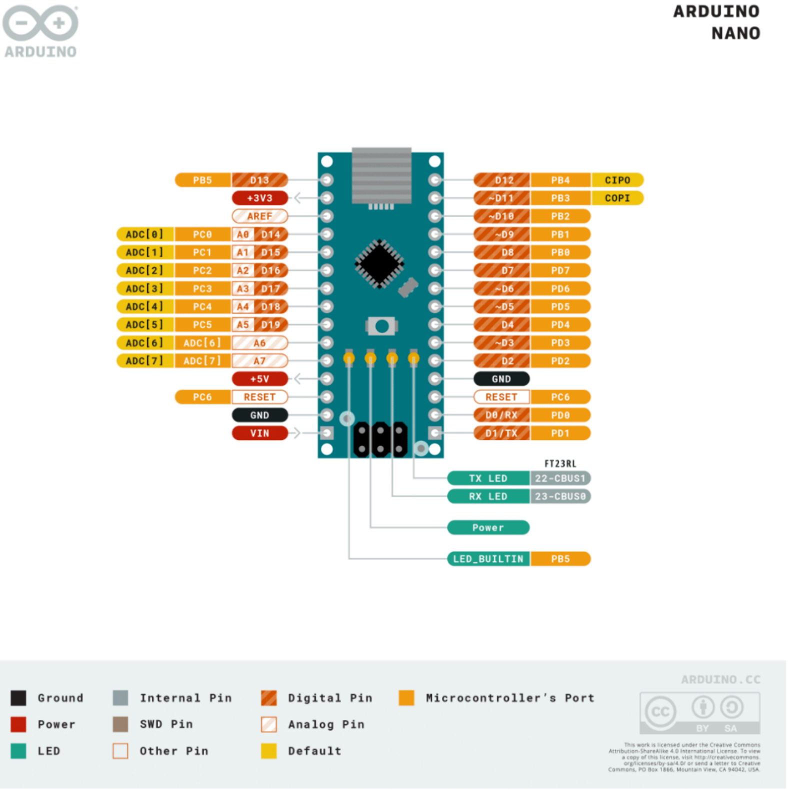
4.3 Sample Sketches

Sample sketches for the Arduino® can be found either in the "Examples" menu in the Arduino® IDE or in the "Documentation" section of the Arduino website **[4]**

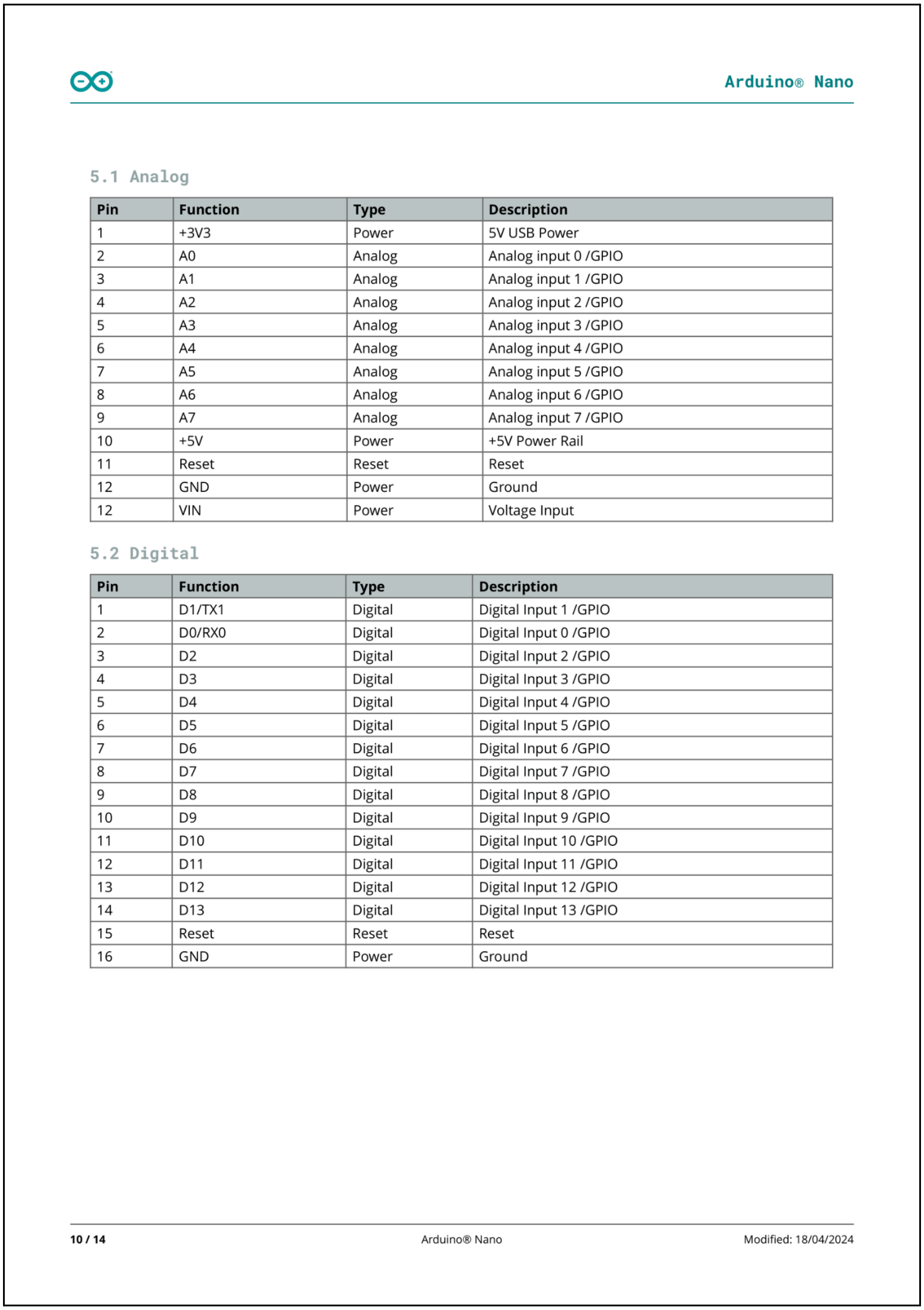
4.4 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on ProjectHub **[5]**, the Arduino® Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more.

5 Connector Pinouts



Power Tree of Arduino Nano



5.1 Analog

Pin	Function	Type	Description
1	+3V3	Power	5V USB Power
2	A0	Analog	Analog input 0 /GPIO
3	A1	Analog	Analog input 1 /GPIO
4	A2	Analog	Analog input 2 /GPIO
5	A3	Analog	Analog input 3 /GPIO
6	A4	Analog	Analog input 4 /GPIO
7	A5	Analog	Analog input 5 /GPIO
8	A6	Analog	Analog input 6 /GPIO
9	A7	Analog	Analog input 7 /GPIO
10	+5V	Power	+5V Power Rail
11	Reset	Reset	Reset
12	GND	Power	Ground
12	VIN	Power	Voltage Input

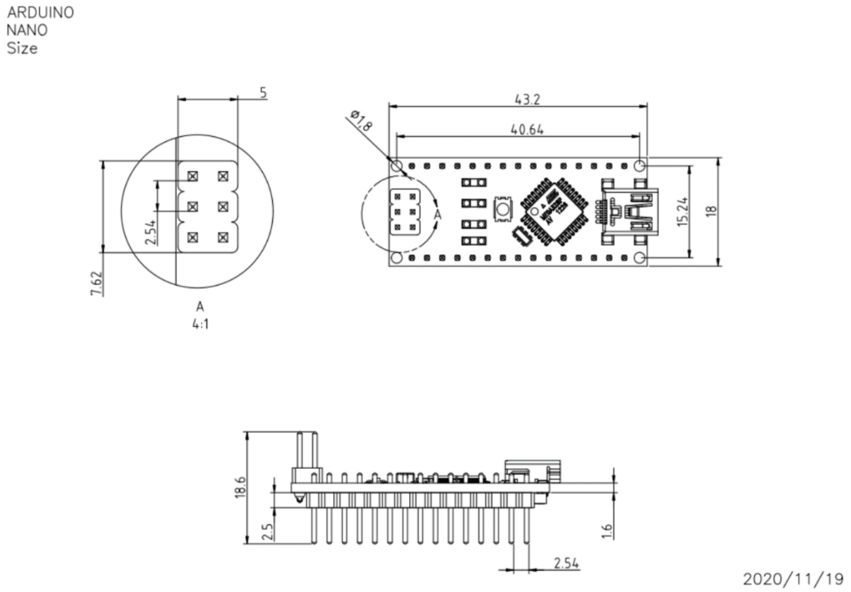
5.2 Digital

Pin	Function	Type	Description
1	D1/TX1	Digital	Digital Input 1 /GPIO
2	D0/RX0	Digital	Digital Input 0 /GPIO
3	D2	Digital	Digital Input 2 /GPIO
4	D3	Digital	Digital Input 3 /GPIO
5	D4	Digital	Digital Input 4 /GPIO
6	D5	Digital	Digital Input 5 /GPIO
7	D6	Digital	Digital Input 6 /GPIO
8	D7	Digital	Digital Input 7 /GPIO
9	D8	Digital	Digital Input 8 /GPIO
10	D9	Digital	Digital Input 9 /GPIO
11	D10	Digital	Digital Input 10 /GPIO
12	D11	Digital	Digital Input 11 /GPIO
13	D12	Digital	Digital Input 12 /GPIO
14	D13	Digital	Digital Input 13 /GPIO
15	Reset	Reset	Reset
16	GND	Power	Ground

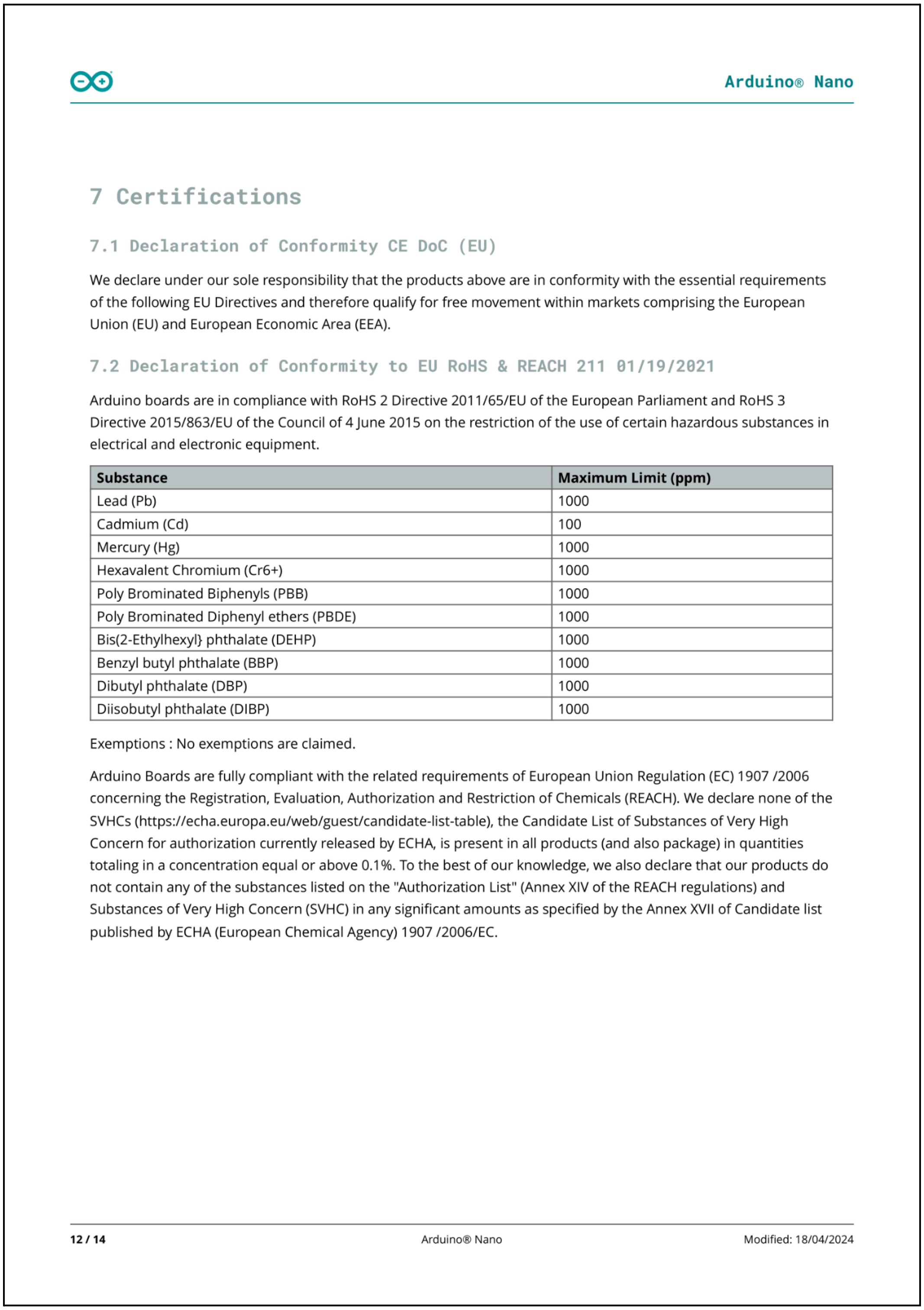
5.3 ATmega328

Pin	Function	Type	Description
1	PB0	Internal	Serial Wire Debug
2	PB1	Internal	Serial Wire Debug
3	PB2	Internal	Serial Wire Debug
4	PB3	Internal	Serial Wire Debug
5	PB4	Internal	Serial Wire Debug
6	PB5	Internal	Serial Wire Debug

6 Mechanical Information



Mechanical dimensions of Arduino Nano



7 Certifications

7.1 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

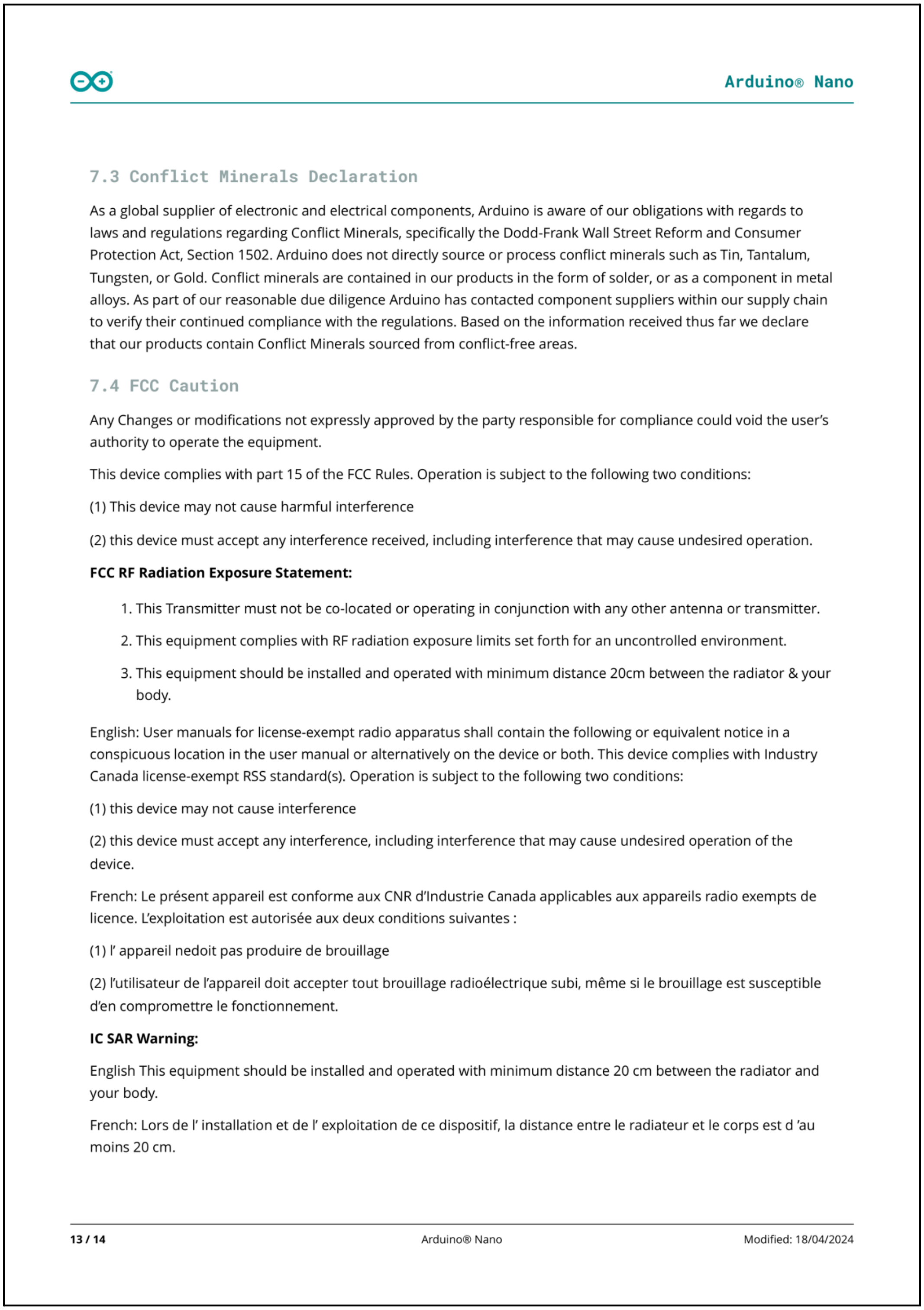
7.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum Limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions : No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.



7.3 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

7.4 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

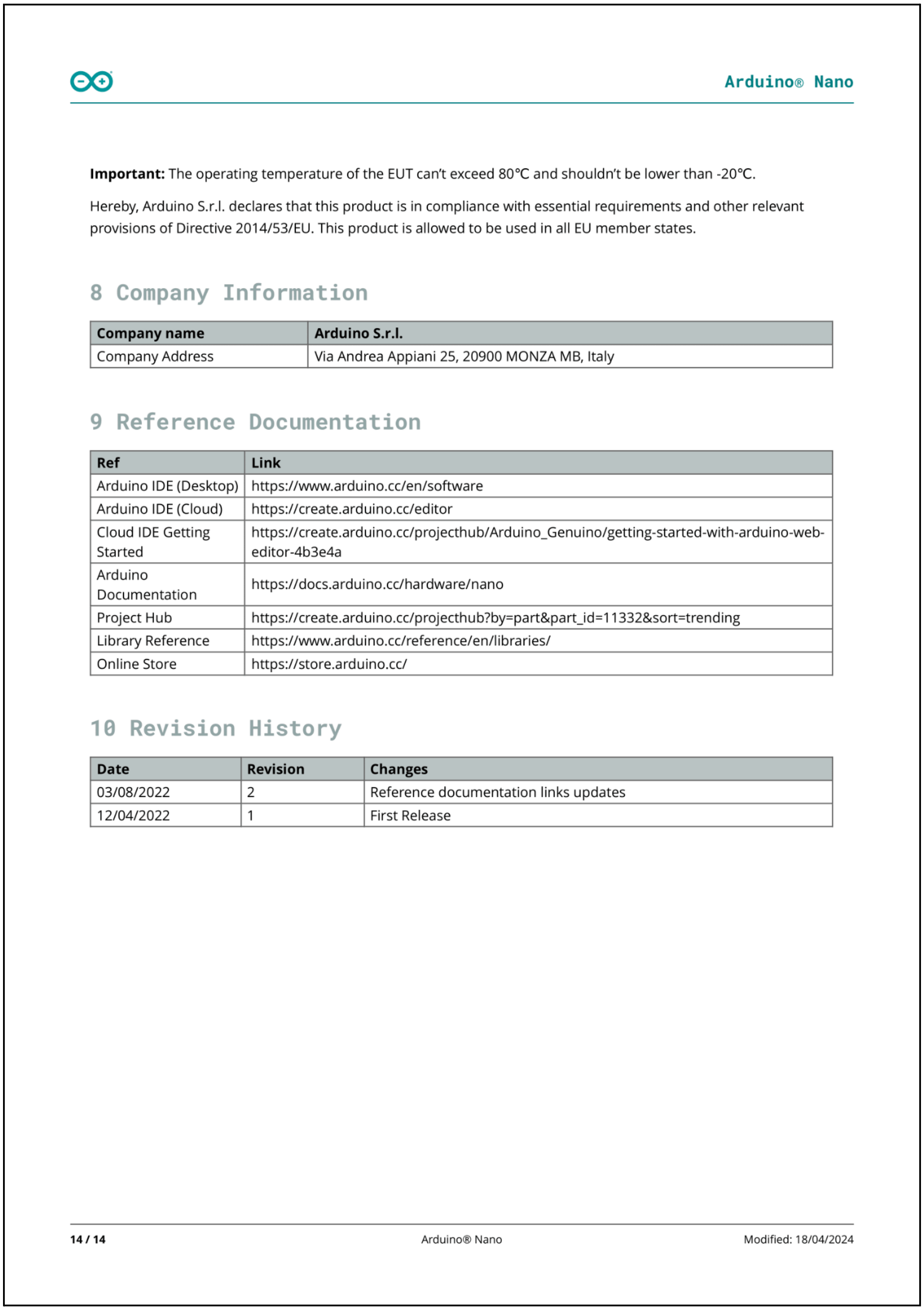
(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.



Important: The operating temperature of the EUT can't exceed 80℃ and shouldn't be lower than -20℃.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 2014/53/EU. This product is allowed to be used in all EU member states.

8 Company Information

Company name	Arduino S.r.l.
Company Address	Via Andrea Appiani 25, 20900 MONZA MB, Italy

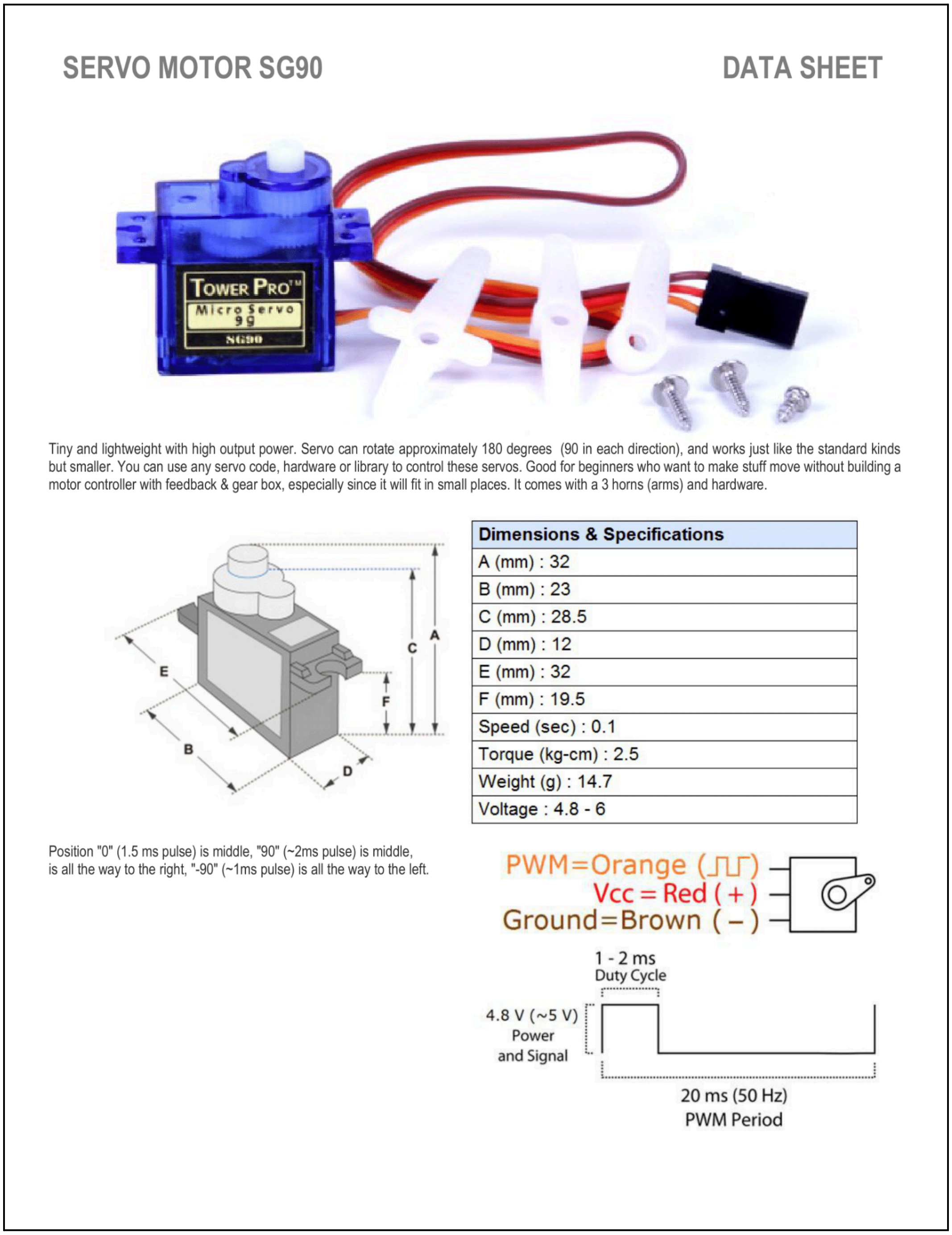
9 Reference Documentation

Ref	Link
Arduino IDE (Desktop)	https://www.arduino.cc/en/software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE Getting Started	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino Documentation	https://docs.arduino.cc/hardware/nano
Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/libraries/
Online Store	https://store.arduino.cc/

10 Revision History

Date	Revision	Changes
03/08/2022	2	Reference documentation links updates
12/04/2022	1	First Release

Tower Pro Servo Motor



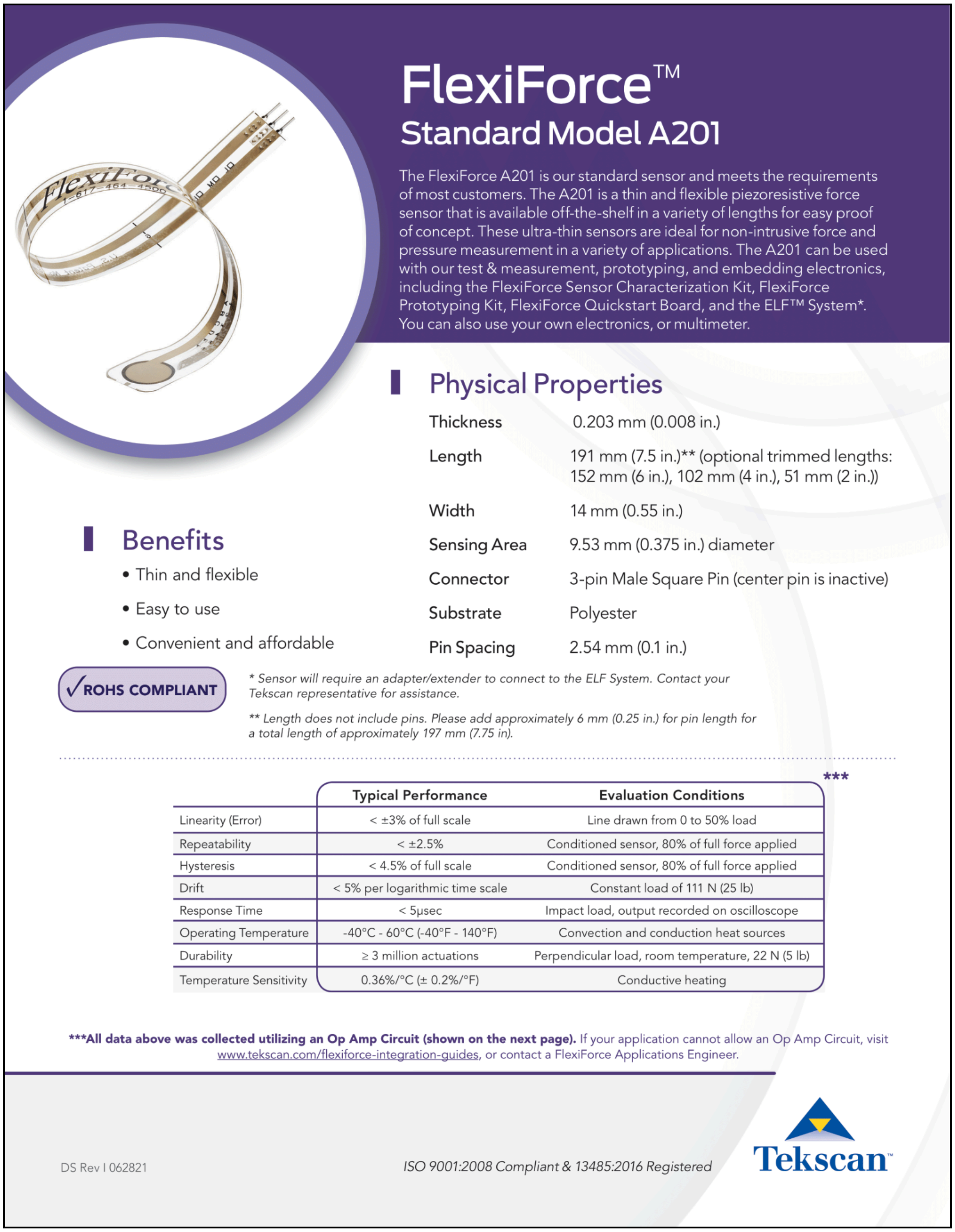
SERVO MOTOR SG90 **DATA SHEET**

Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.

Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.

Flexiforce Strain Gauge



FlexiForce™

Standard Model A201

The FlexiForce A201 is our standard sensor and meets the requirements of most customers. The A201 is a thin and flexible piezoresistive force sensor that is available off-the-shelf in a variety of lengths for easy proof of concept. These ultra-thin sensors are ideal for non-intrusive force and pressure measurement in a variety of applications. The A201 can be used with our test & measurement, prototyping, and embedding electronics, including the FlexiForce Sensor Characterization Kit, FlexiForce Prototyping Kit, FlexiForce Quickstart Board, and the ELF™ System*. You can also use your own electronics, or multimeter.

Physical Properties

Thickness	0.203 mm (0.008 in.)
Length	191 mm (7.5 in.)** (optional trimmed lengths: 152 mm (6 in.), 102 mm (4 in.), 51 mm (2 in.))
Width	14 mm (0.55 in.)
Sensing Area	9.53 mm (0.375 in.) diameter
Connector	3-pin Male Square Pin (center pin is inactive)
Substrate	Polyester
Pin Spacing	2.54 mm (0.1 in.)

Benefits

- Thin and flexible
- Easy to use
- Convenient and affordable

✓ROHS COMPLIANT

** Sensor will require an adapter/extender to connect to the ELF System. Contact your Tekscan representative for assistance.*

*** Length does not include pins. Please add approximately 6 mm (0.25 in.) for pin length for a total length of approximately 197 mm (7.75 in).*

***

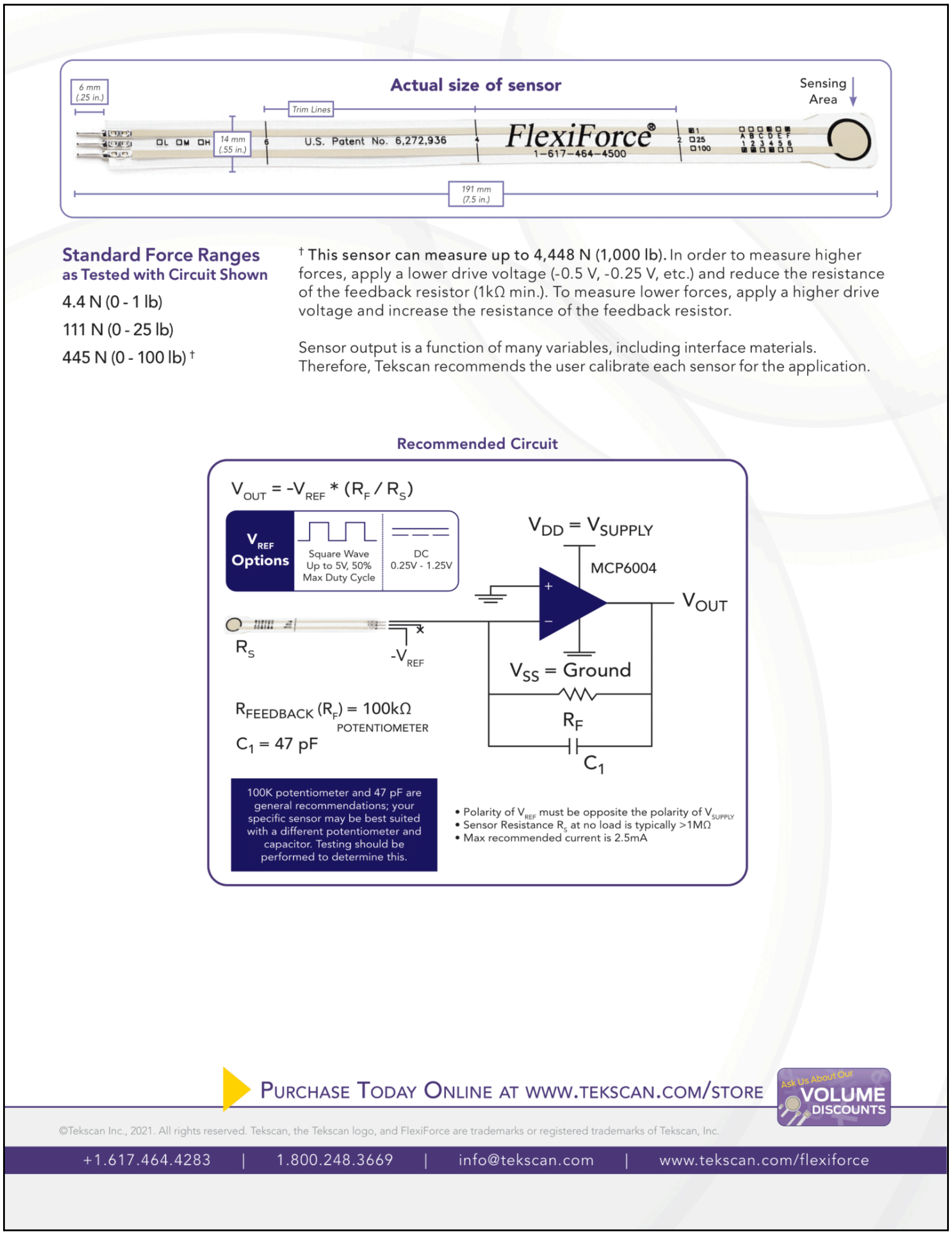
	Typical Performance	Evaluation Conditions
Linearity (Error)	< ±3% of full scale	Line drawn from 0 to 50% load
Repeatability	< ±2.5%	Conditioned sensor, 80% of full force applied
Hysteresis	< 4.5% of full scale	Conditioned sensor, 80% of full force applied
Drift	< 5% per logarithmic time scale	Constant load of 111 N (25 lb)
Response Time	< 5μsec	Impact load, output recorded on oscilloscope
Operating Temperature	-40°C - 60°C (-40°F - 140°F)	Convection and conduction heat sources
Durability	≥ 3 million actuations	Perpendicular load, room temperature, 22 N (5 lb)
Temperature Sensitivity	0.36%/°C (± 0.2%/°F)	Conductive heating

*****All data above was collected utilizing an Op Amp Circuit (shown on the next page).** If your application cannot allow an Op Amp Circuit, visit www.tekscan.com/flexiforce-integration-guides, or contact a FlexiForce Applications Engineer.

DS Rev I 062821

ISO 9001:2008 Compliant & 13485:2016 Registered

Tekscan™



Actual size of sensor

Standard Force Ranges
as Tested with Circuit Shown

4.4 N (0 - 1 lb)

111 N (0 - 25 lb)

445 N (0 - 100 lb) †

† **This sensor can measure up to 4,448 N (1,000 lb).** In order to measure higher forces, apply a lower drive voltage (-0.5 V, -0.25 V, etc.) and reduce the resistance of the feedback resistor (1kΩ min.). To measure lower forces, apply a higher drive voltage and increase the resistance of the feedback resistor.

Sensor output is a function of many variables, including interface materials. Therefore, Tekscan recommends the user calibrate each sensor for the application.

Recommended Circuit

$V_{OUT} = -V_{REF} * (R_F / R_S)$

V_{REF} Options

$V_{DD} = V_{SUPPLY}$

V_{SS} = Ground

$R_{FEEDBACK}$ (R_F) = 100kΩ POTENTIOMETER

C_1 = 47 pF

100K potentiometer and 47 pF are general recommendations; your specific sensor may be best suited with a different potentiometer and capacitor. Testing should be performed to determine this.

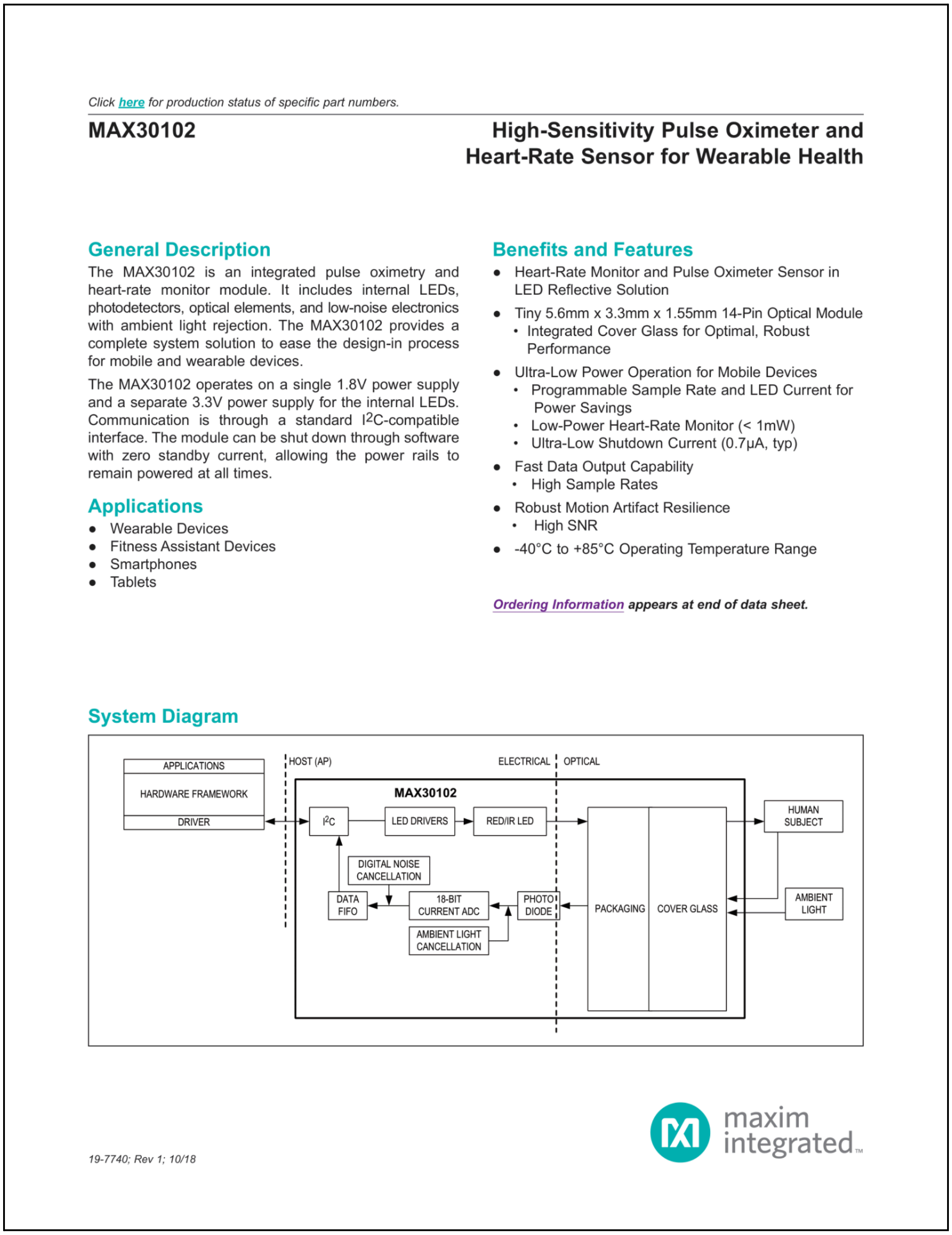
- Polarity of V_{REF} must be opposite the polarity of V_{SUPPLY}
- Sensor Resistance R_S at no load is typically >1MΩ
- Max recommended current is 2.5mA

PURCHASE TODAY ONLINE AT WWW.TEKSCAN.COM/STORE

Ask Us About Our VOLUME DISCOUNTS

+1.617.464.4283 | 1.800.248.3669 | info@tekscan.com | www.tekscan.com/flexiforce

Max 30102 Pulse Oximeter



*Click **here** for production status of specific part numbers.*

MAX30102

High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health

General Description

The MAX30102 is an integrated pulse oximetry and heart-rate monitor module. It includes internal LEDs, photodetectors, optical elements, and low-noise electronics with ambient light rejection. The MAX30102 provides a complete system solution to ease the design-in process for mobile and wearable devices.

The MAX30102 operates on a single 1.8V power supply and a separate 3.3V power supply for the internal LEDs. Communication is through a standard I^2C-compatible interface. The module can be shut down through software with zero standby current, allowing the power rails to remain powered at all times.

Applications

- Wearable Devices
- Fitness Assistant Devices
- Smartphones
- Tablets

Benefits and Features

- Heart-Rate Monitor and Pulse Oximeter Sensor in LED Reflective Solution
- Tiny 5.6mm x 3.3mm x 1.55mm 14-Pin Optical Module
  - Integrated Cover Glass for Optimal, Robust Performance
- Ultra-Low Power Operation for Mobile Devices
  - Programmable Sample Rate and LED Current for Power Savings
  - Low-Power Heart-Rate Monitor (< 1mW)
  - Ultra-Low Shutdown Current (0.7µA, typ)
- Fast Data Output Capability
  - High Sample Rates
- Robust Motion Artifact Resilience
  - High SNR
- -40°C to +85°C Operating Temperature Range

***Ordering Information** appears at end of data sheet.*

System Diagram

maxim integrated

19-7740; Rev 1; 10/18

Absolute Maximum Ratings

V_{DD} to GND	-0.3V to +2.2V
GND to PGND	-0.3V to +0.3V
V_{LED+} to PGND	-0.3V to +6.0V
All Other Pins to GND	-0.3V to +6.0V
Output Short-Circuit Current Duration	Continuous
Continuous Input Current into Any Terminal	±20mA
ESD, Human Body Model (HBM)	2.5kV
Latchup Immunity	±250mA
Continuous Power Dissipation (T_A = +70°C)	
OESIP (derate 5.5mW/°C above +70°C)	440mW
Operating Temperature Range	-40°C to +85°C
Junction Temperature	+90°C
Soldering Temperature (reflow)	+260°C
Storage Temperature Range	-40°C to +105°C

Package Information

PACKAGE TYPE: 14 OESIP	
Package Code	F143A5MK+1
Outline Number	**21-1048**
Land Pattern Number	**90-0602**
THERMAL RESISTANCE, FOUR-LAYER BOARD	
Junction to Ambient (θ_{JA})	180°C/W
Junction to Case (θ_{JC})	150°C/W

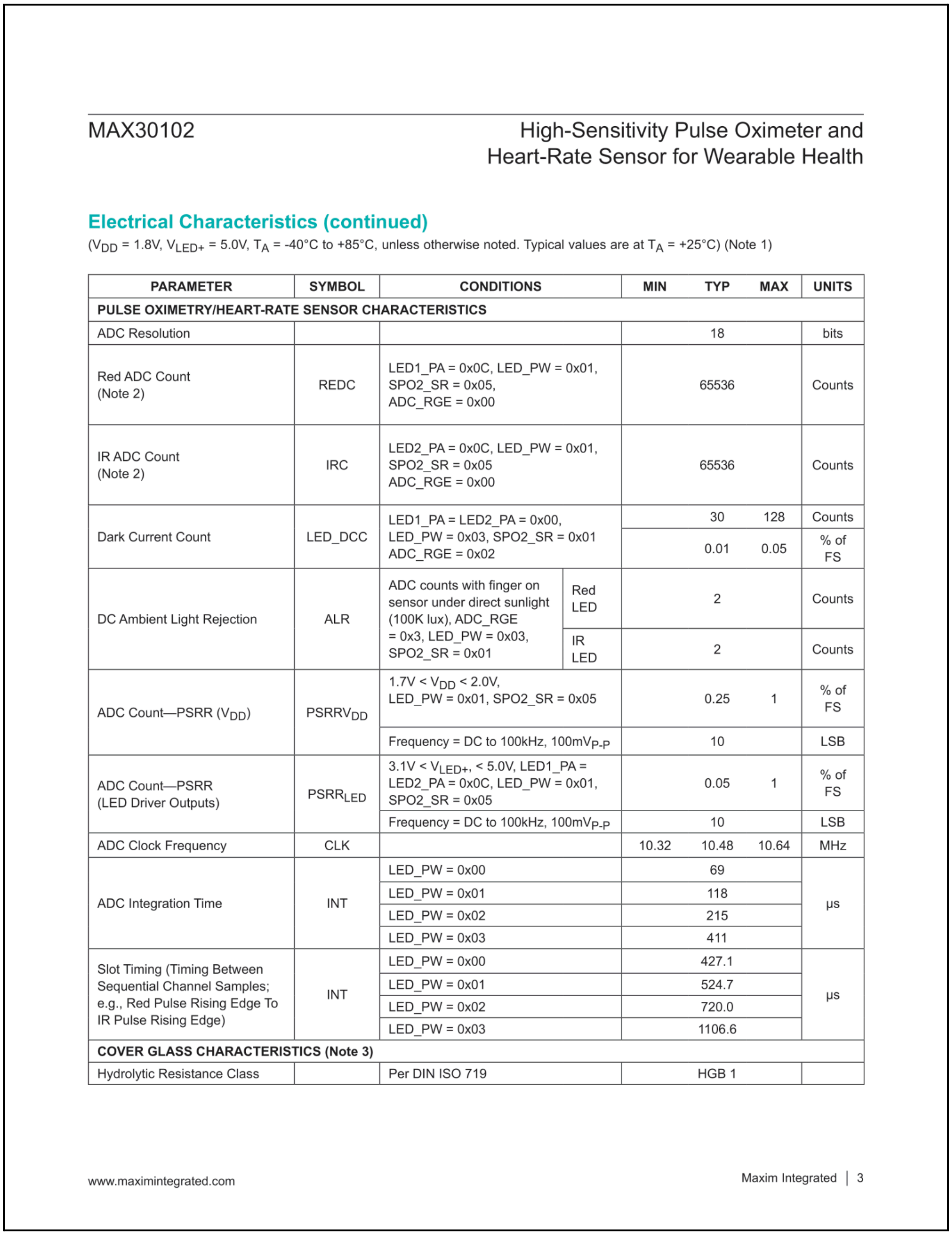
Package thermal resistances were obtained using the method described in JEDEC specification JESD51-7, using a four-layer board. For detailed information on package thermal considerations, refer to **www.maximintegrated.com/thermal-tutorial**.

For the latest package outline information and land patterns (footprints), go to **www.maximintegrated.com/packages**. Note that a "+", "#", or "-" in the package code indicates RoHS status only. Package drawings may show a different suffix character, but the drawing pertains to the package regardless of RoHS status.

Electrical Characteristics

(V_{DD} = 1.8V, V_{LED+} = 5.0V, T_A = -40°C to +85°C, unless otherwise noted. Typical values are at T_A = +25°C) (Note 1)

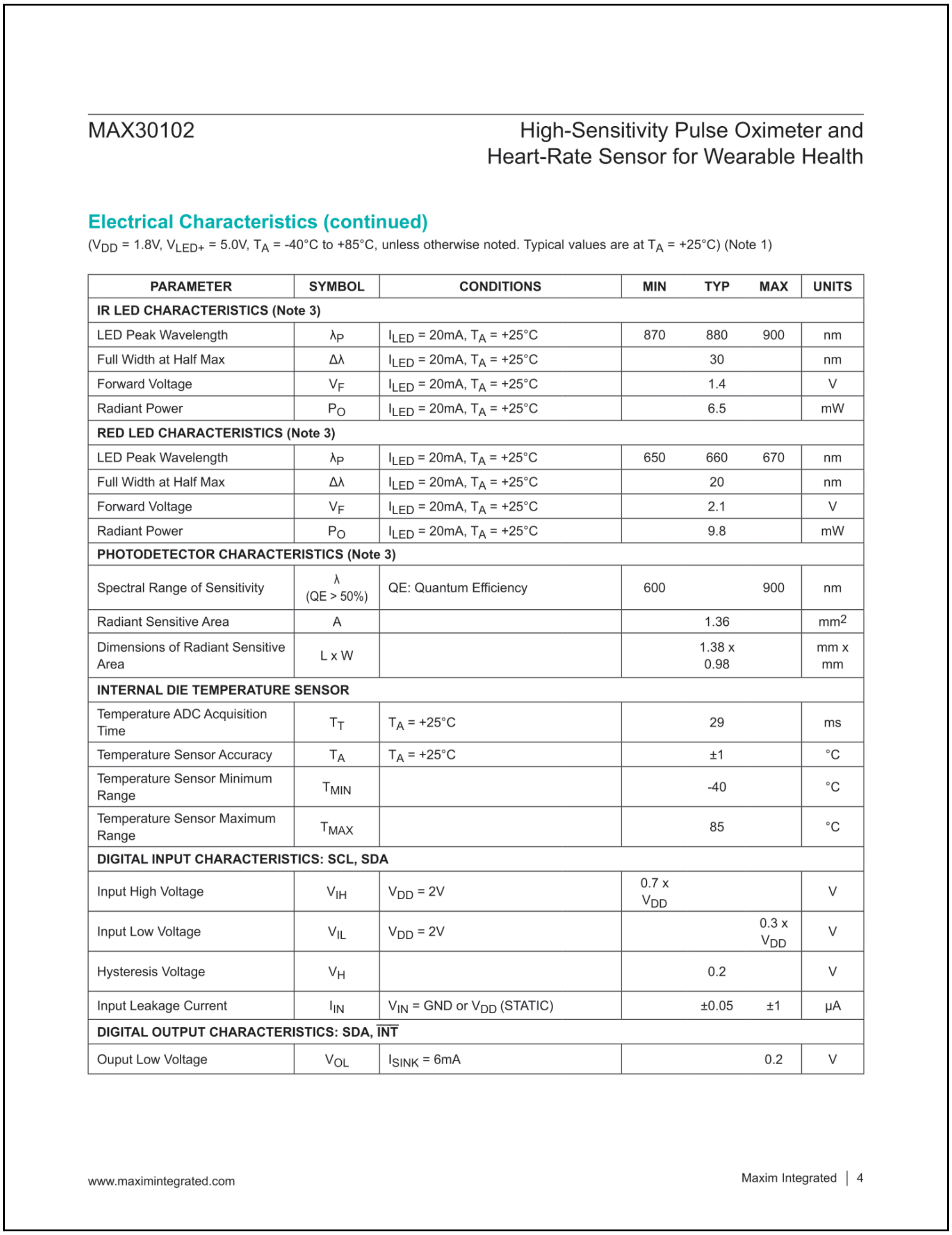
PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
POWER SUPPLY						
Power-Supply Voltage	V_{DD}	Guaranteed by RED and IR count tolerance	1.7	1.8	2.0	V
LED Supply Voltage V_{LED+} to PGND	V_{LED+}	Guaranteed by PSRR of LED driver	3.1	3.3	5.0	V
Supply Current	I_{DD}	SpO_2 and HR mode, PW = 215µs, 50sps		600	1200	µA
		IR only mode, PW = 215µS, 50sps		600	1200	
Supply Current in Shutdown	I_{SHDN}	T_A = +25°C, MODE = 0x80		0.7	10	µA



Electrical Characteristics (continued)

(V_{DD} = 1.8V, V_{LED+} = 5.0V, T_A = -40°C to +85°C, unless otherwise noted. Typical values are at T_A = +25°C) (Note 1)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
PULSE OXIMETRY/HEART-RATE SENSOR CHARACTERISTICS							
ADC Resolution					18		bits
Red ADC Count (Note 2)	REDC	LED1_PA = 0x0C, LED_PW = 0x01, SPO2_SR = 0x05, ADC_RGE = 0x00			65536		Counts
IR ADC Count (Note 2)	IRC	LED2_PA = 0x0C, LED_PW = 0x01, SPO2_SR = 0x05 ADC_RGE = 0x00			65536		Counts
Dark Current Count	LED_DCC	LED1_PA = LED2_PA = 0x00, LED_PW = 0x03, SPO2_SR = 0x01 ADC_RGE = 0x02			30	128	Counts
					0.01	0.05	% of FS
DC Ambient Light Rejection	ALR	ADC counts with finger on sensor under direct sunlight (100K lux), ADC_RGE = 0x3, LED_PW = 0x03, SPO2_SR = 0x01	Red LED		2		Counts
			IR LED		2		Counts
ADC Count—PSRR (V_{DD})	$PSRRV_{DD}$	1.7V < V_{DD} < 2.0V, LED_PW = 0x01, SPO2_SR = 0x05			0.25	1	% of FS
		Frequency = DC to 100kHz, $100mV_{P-P}$			10		LSB
ADC Count—PSRR (LED Driver Outputs)	$PSRR_{LED}$	3.1V < V_{LED+}, < 5.0V, LED1_PA = LED2_PA = 0x0C, LED_PW = 0x01, SPO2_SR = 0x05			0.05	1	% of FS
		Frequency = DC to 100kHz, $100mV_{P-P}$			10		LSB
ADC Clock Frequency	CLK			10.32	10.48	10.64	MHz
ADC Integration Time	INT	LED_PW = 0x00			69		µs
		LED_PW = 0x01			118		
		LED_PW = 0x02			215		
		LED_PW = 0x03			411		
Slot Timing (Timing Between Sequential Channel Samples; e.g., Red Pulse Rising Edge To IR Pulse Rising Edge)	INT	LED_PW = 0x00			427.1		µs
		LED_PW = 0x01			524.7		
		LED_PW = 0x02			720.0		
		LED_PW = 0x03			1106.6		
COVER GLASS CHARACTERISTICS (Note 3)							
Hydrolytic Resistance Class		Per DIN ISO 719			HGB 1		



MAX30102

High-Sensitivity Pulse Oximeter and
Heart-Rate Sensor for Wearable Health

Electrical Characteristics (continued)

(V_{DD} = 1.8V, V_{LED+} = 5.0V, T_A = -40°C to +85°C, unless otherwise noted. Typical values are at T_A = +25°C) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
IR LED CHARACTERISTICS (Note 3)						
LED Peak Wavelength	λ_P	I_{LED} = 20mA, T_A = +25°C	870	880	900	nm
Full Width at Half Max	$\Delta\lambda$	I_{LED} = 20mA, T_A = +25°C		30		nm
Forward Voltage	V_F	I_{LED} = 20mA, T_A = +25°C		1.4		V
Radiant Power	P_O	I_{LED} = 20mA, T_A = +25°C		6.5		mW
RED LED CHARACTERISTICS (Note 3)						
LED Peak Wavelength	λ_P	I_{LED} = 20mA, T_A = +25°C	650	660	670	nm
Full Width at Half Max	$\Delta\lambda$	I_{LED} = 20mA, T_A = +25°C		20		nm
Forward Voltage	V_F	I_{LED} = 20mA, T_A = +25°C		2.1		V
Radiant Power	P_O	I_{LED} = 20mA, T_A = +25°C		9.8		mW
PHOTODETECTOR CHARACTERISTICS (Note 3)						
Spectral Range of Sensitivity	λ (QE > 50%)	QE: Quantum Efficiency	600		900	nm
Radiant Sensitive Area	A			1.36		mm^2
Dimensions of Radiant Sensitive Area	L x W			1.38 x 0.98		mm x mm
INTERNAL DIE TEMPERATURE SENSOR						
Temperature ADC Acquisition Time	T_T	T_A = +25°C		29		ms
Temperature Sensor Accuracy	T_A	T_A = +25°C		±1		°C
Temperature Sensor Minimum Range	T_{MIN}			-40		°C
Temperature Sensor Maximum Range	T_{MAX}			85		°C
DIGITAL INPUT CHARACTERISTICS: SCL, SDA						
Input High Voltage	V_{IH}	V_{DD} = 2V	0.7 x V_{DD}			V
Input Low Voltage	V_{IL}	V_{DD} = 2V			0.3 x V_{DD}	V
Hysteresis Voltage	V_H			0.2		V
Input Leakage Current	I_{IN}	V_{IN} = GND or V_{DD} (STATIC)		±0.05	±1	µA
DIGITAL OUTPUT CHARACTERISTICS: SDA, $\overline{INT}$						
Ouput Low Voltage	V_{OL}	I_{SINK} = 6mA			0.2	V

Electrical Characteristics (continued)

(V_{DD} = 1.8V, V_{LED+} = 5.0V, T_A = -40°C to +85°C, unless otherwise noted. Typical values are at T_A = +25°C) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
I^{2}C TIMING CHARACTERISTICS (SDA, SDA, $\overline{\text{INT}}$) (Note 3)						
I^{2}C Write Address				AE		Hex
I^{2}C Read Address				AF		Hex
Serial Clock Frequency	f_{SCL}		0		400	kHz
Bus Free Time Between STOP and START Conditions	t_{BUF}		1.3			µs
Hold Time (Repeated) START Condition	$t_{HD;STA}$		0.6			µs
SCL Pulse-Width Low	t_{LOW}		1.3			µs
SCL Pulse-Width High	t_{HIGH}		0.6			µs
Setup Time for a Repeated START Condition	$t_{SU;STA}$		0.6			µs
Data Hold Time	$t_{HD;DAT}$		0		900	ns
Data Setup Time	$t_{SU;DAT}$		100			ns
Setup Time for STOP Condition	$t_{SU;STO}$		0.6			µs
Pulse Width of Suppressed Spike	t_{SP}		0		50	ns
Bus Capacitance	C_B				400	pF
SDA and SCL Receiving Rise Time	t_R		$20 + 0.1C_B$		300	ns
SDA and SCL Receiving Fall Time	t_{RF}		$20 + 0.1C_B$		300	ns
SDA Transmitting Fall Time	t_{TF}				300	ns

Note 1: All devices are 100% production tested at T_A = +25°C. Specifications over temperature limits are guaranteed by Maxim Integrated's bench or proprietary automated test equipment (ATE) characterization.

Note 2: Specifications are guaranteed by Maxim Integrated's bench characterization and by 100% production test using proprietary ATE setup and conditions.

Note 3: Guaranteed by design and characterization. Not tested in final production.

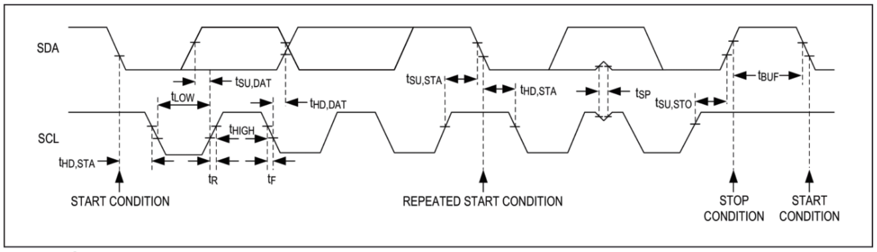
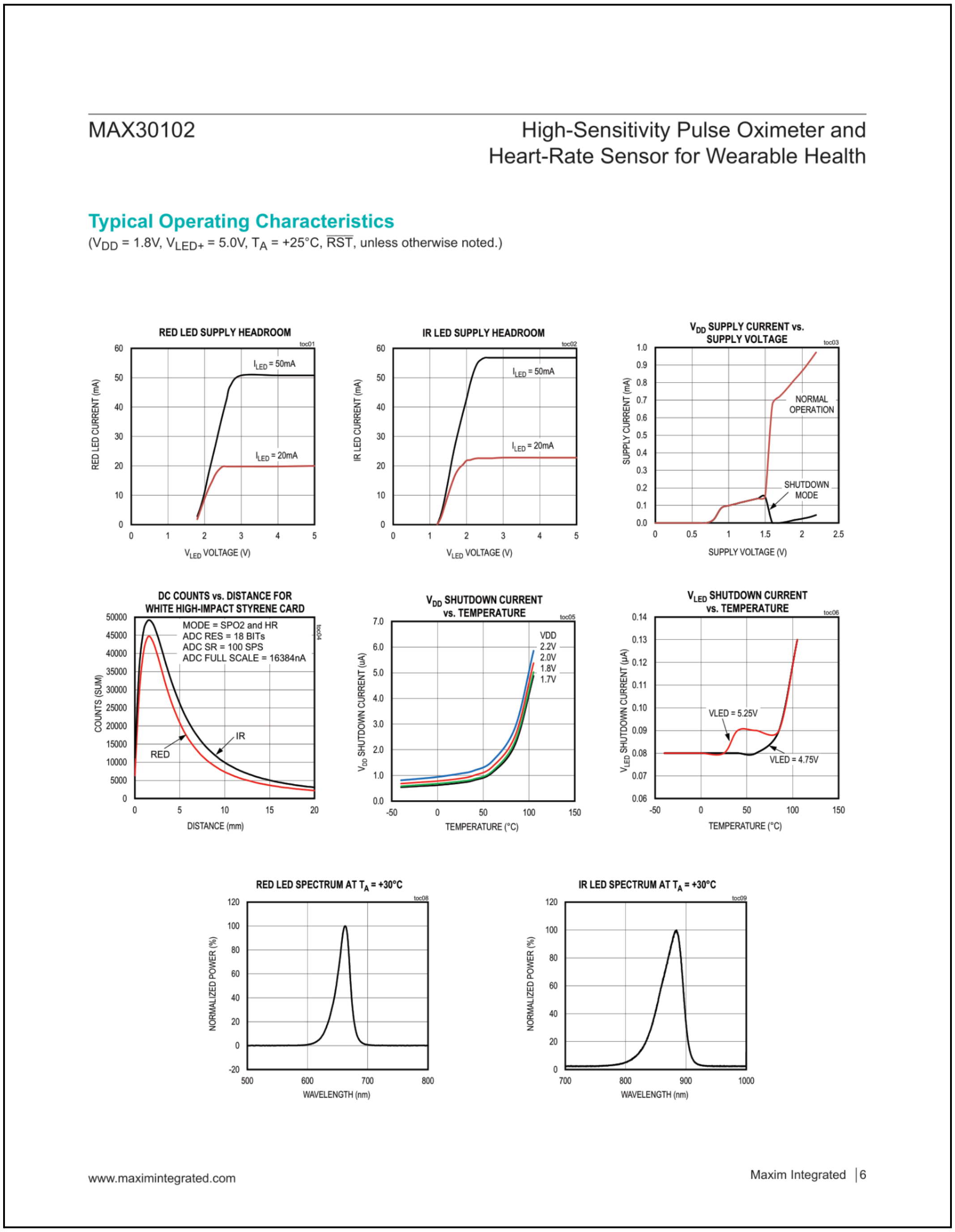


Figure 1. I^{2}C-Compatible Interface Timing Diagram



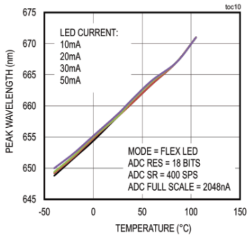
Typical Operating Characteristics

(V_{DD} = 1.8V, V_{LED+} = 5.0V, T_A = +25°C, $\overline{RST}$, unless otherwise noted.)

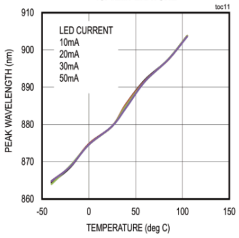
Typical Operating Characteristics (continued)

(V_{DD} = 1.8V, V_{LED+} = 5.0V, T_A = +25°C, $\overline{RST}$, unless otherwise noted.)

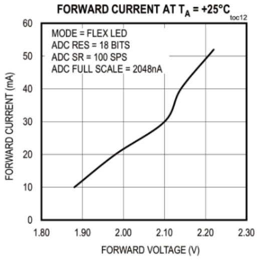
RED LED PEAK WAVELENGTH vs. TEMPERATURE



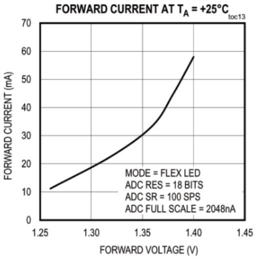
IR LED PEAK WAVELENGTH vs. TEMPERATURE



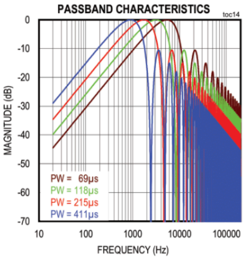
RED LED FORWARD VOLTAGE vs. FORWARD CURRENT AT T_A = +25°C



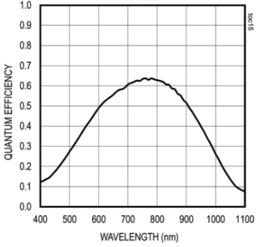
IR LED FORWARD VOLTAGE vs. FORWARD CURRENT AT T_A = +25°C

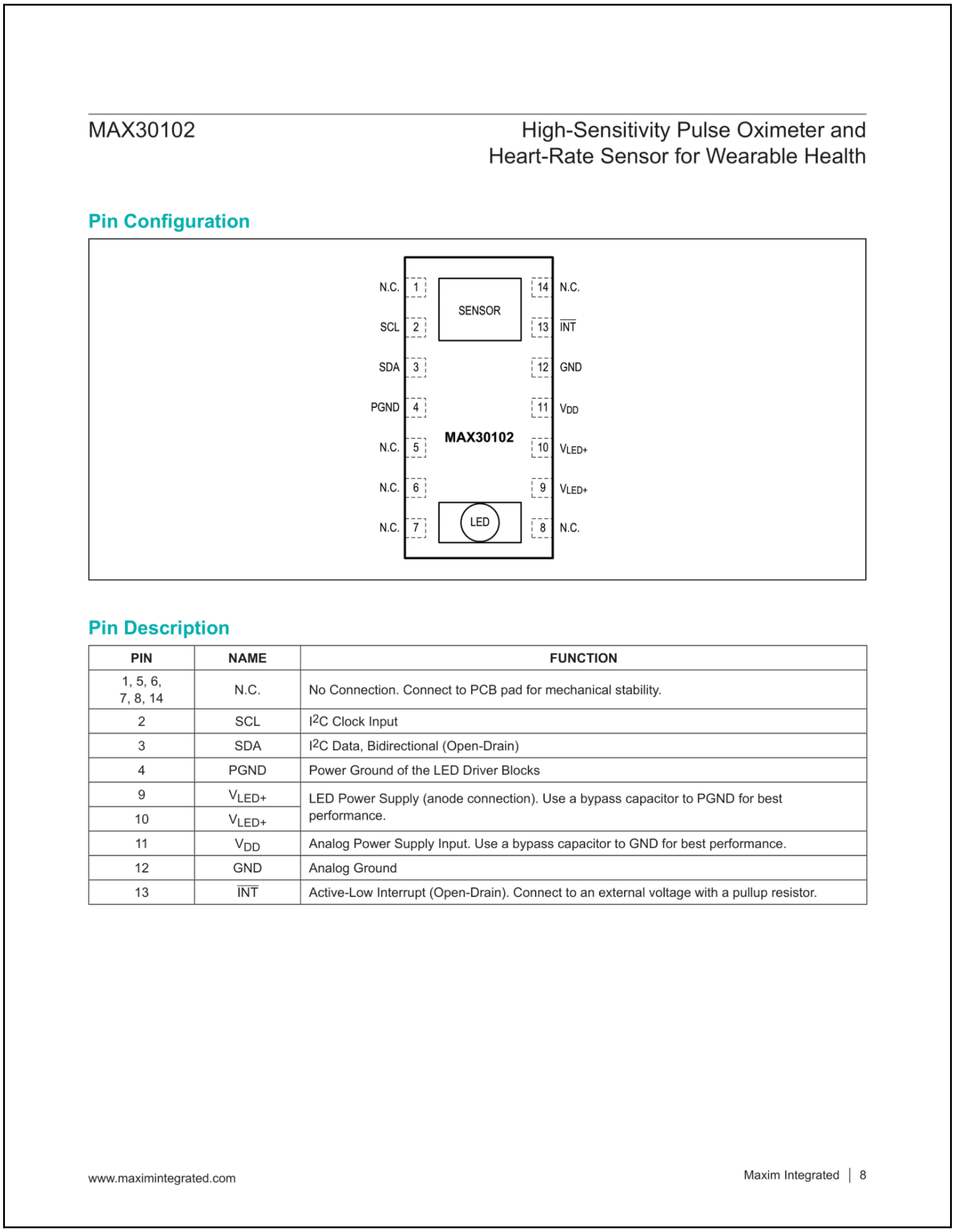


AMBIENT LIGHT CANCELLATION PASSBAND CHARACTERISTICS



PHOTODIODE QUANTUM EFFICIENCY vs. WAVELENGTH



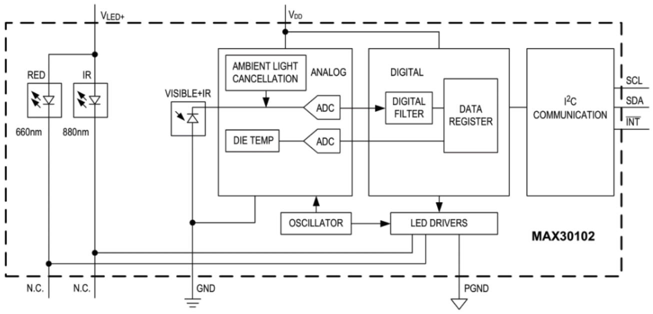


Pin Configuration

Pin Description

PIN	NAME	FUNCTION
1, 5, 6, 7, 8, 14	N.C.	No Connection. Connect to PCB pad for mechanical stability.
2	SCL	I^2C Clock Input
3	SDA	I^2C Data, Bidirectional (Open-Drain)
4	PGND	Power Ground of the LED Driver Blocks
9	V_{LED+}	LED Power Supply (anode connection). Use a bypass capacitor to PGND for best performance.
10	V_{LED+}	
11	V_{DD}	Analog Power Supply Input. Use a bypass capacitor to GND for best performance.
12	GND	Analog Ground
13	$\overline{\text{INT}}$	Active-Low Interrupt (Open-Drain). Connect to an external voltage with a pullup resistor.

Functional Diagram



Detailed Description

The MAX30102 is a complete pulse oximetry and heart-rate sensor system solution module designed for the demanding requirements of wearable devices. The device maintains a very small solution size without sacrificing optical or electrical performance. Minimal external hardware components are required for integration into a wearable system.

The MAX30102 is fully adjustable through software registers, and the digital output data can be stored in a 32-deep FIFO within the IC. The FIFO allows the MAX30102 to be connected to a microcontroller or processor on a shared bus, where the data is not being read continuously from the MAX30102's registers.

SpO_2 Subsystem

The SpO_2 subsystem of the MAX30102 contains ambient light cancellation (ALC), a continuous-time sigma-delta ADC, and a proprietary discrete time filter. The ALC has an internal Track/Hold circuit to cancel ambient light and increase the effective dynamic range. The SpO_2 ADC has programmable full-scale ranges from 2μA to 16μA. The ALC can cancel up to 200μA of ambient current.

The internal ADC is a continuous time oversampling sigma-delta converter with 18-bit resolution. The ADC sampling rate is 10.24MHz. The ADC output data rate can be programmed from 50sps (samples per second) to 3200sps.

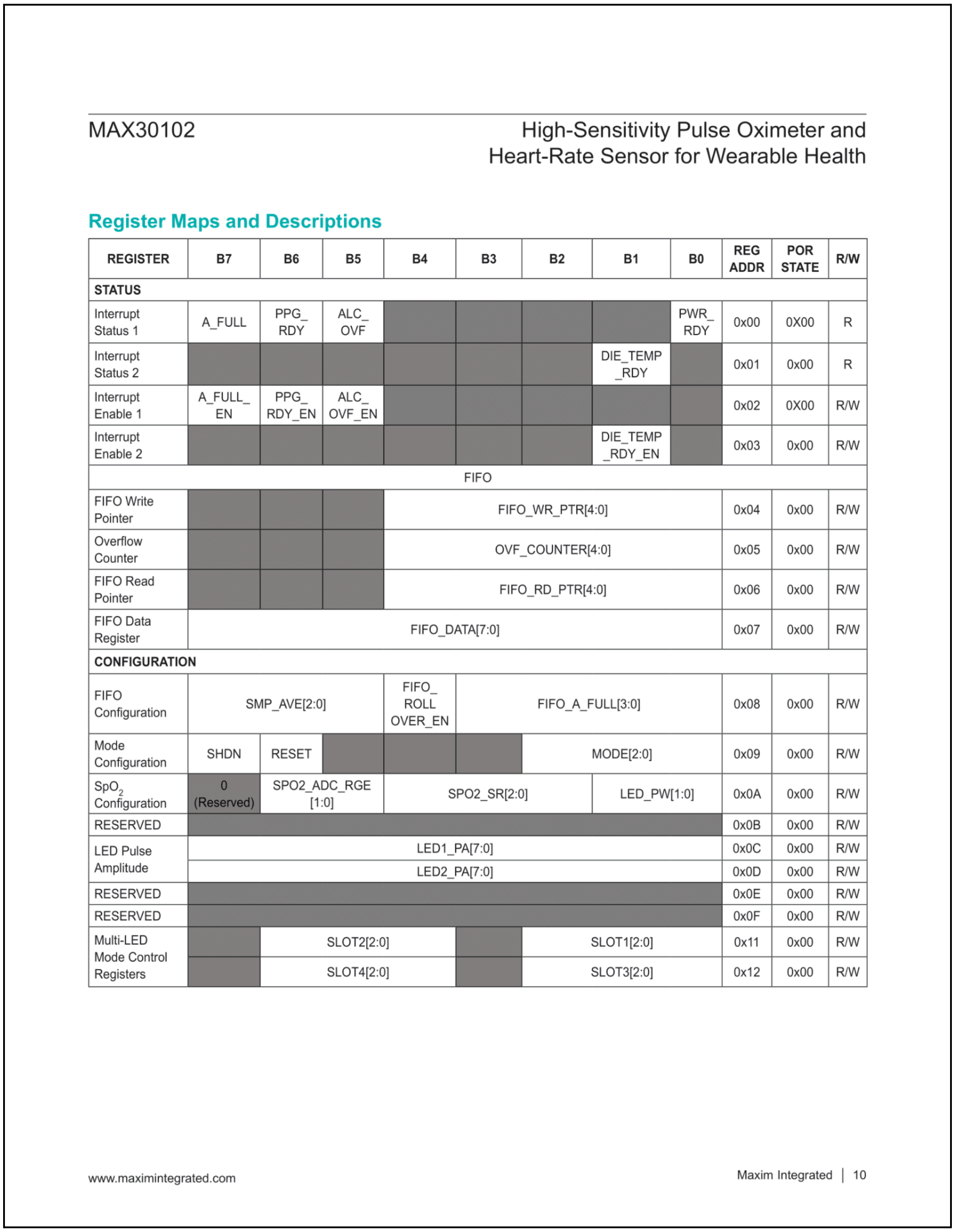
Temperature Sensor

The MAX30102 has an on-chip temperature sensor for calibrating the temperature dependence of the SpO_2 subsystem. The temperature sensor has an inherent resolution of 0.0625°C.

The device output data is relatively insensitive to the wavelength of the IR LED, where the Red LED's wavelength is critical to correct interpretation of the data. An SpO_2 algorithm used with the MAX30102 output signal can compensate for the associated SpO_2 error with ambient temperature changes.

LED Driver

The MAX30102 integrates Red and IR LED drivers to modulate LED pulses for SpO_2 and HR measurements. The LED current can be programmed from 0 to 50mA with proper supply voltage. The LED pulse width can be programmed from 69μs to 411μs to allow the algorithm to optimize SpO_2 and HR accuracy and power consumption based on use cases.



MAX30102

High-Sensitivity Pulse Oximeter and
Heart-Rate Sensor for Wearable Health

Register Maps and Descriptions

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
STATUS											
Interrupt Status 1	A_FULL	PPG_ RDY	ALC_ OVF					PWR_ RDY	0x00	0X00	R
Interrupt Status 2							DIE_TEMP _RDY		0x01	0x00	R
Interrupt Enable 1	A_FULL_ EN	PPG_ RDY_EN	ALC_ OVF_EN						0x02	0X00	R/W
Interrupt Enable 2							DIE_TEMP _RDY_EN		0x03	0x00	R/W
FIFO											
FIFO Write Pointer				FIFO_WR_PTR[4:0]					0x04	0x00	R/W
Overflow Counter				OVF_COUNTER[4:0]					0x05	0x00	R/W
FIFO Read Pointer				FIFO_RD_PTR[4:0]					0x06	0x00	R/W
FIFO Data Register	FIFO_DATA[7:0]								0x07	0x00	R/W
CONFIGURATION											
FIFO Configuration	SMP_AVE[2:0]			FIFO_ ROLL OVER_EN	FIFO_A_FULL[3:0]				0x08	0x00	R/W
Mode Configuration	SHDN	RESET				MODE[2:0]			0x09	0x00	R/W
SpO_2 Configuration	0 (Reserved)	SPO2_ADC_RGE [1:0]		SPO2_SR[2:0]			LED_PW[1:0]		0x0A	0x00	R/W
RESERVED									0x0B	0x00	R/W
LED Pulse Amplitude	LED1_PA[7:0]								0x0C	0x00	R/W
	LED2_PA[7:0]								0x0D	0x00	R/W
RESERVED									0x0E	0x00	R/W
RESERVED									0x0F	0x00	R/W
Multi-LED Mode Control Registers		SLOT2[2:0]				SLOT1[2:0]			0x11	0x00	R/W
		SLOT4[2:0]				SLOT3[2:0]			0x12	0x00	R/W

www.maximintegrated.com

Maxim Integrated │ 10

Register Maps and Descriptions (continued)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
RESERVED									0x13–0x17	0xFF	R/W
RESERVED									0x18-0x1E	0x00	R
DIE TEMPERATURE											
Die Temp Integer	TINT[7:0]								0x1F	0x00	R
Die Temp Fraction					TFRAC[3:0]				0x20	0x00	R
Die Temperature Config								TEMP_EN	0x21	0x00	R/W
RESERVED									0x22–0x2F	0x00	R/W
PART ID											
Revision ID	REV_ID[7:0]								0xFE	0xXX*	R
Part ID	PART_ID[7]								0xFF	0x15	R

**XX denotes a 2-digit hexadecimal number (00 to FF) for part revision identification. Contact Maxim Integrated for the revision ID number assigned for your product.*

Interrupt Status (0x00–0x01)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
Interrupt Status 1	A_FULL	PPG_RDY	ALC_OVF					PWR_ RDY	0x00	0X00	R
Interrupt Status 2							DIE_ TEMP_RDY		0x01	0x00	R

Whenever an interrupt is triggered, the MAX30102 pulls the active-low interrupt pin into its low state until the interrupt is cleared.

A_FULL: FIFO Almost Full Flag

In SpO_2 and HR modes, this interrupt triggers when the FIFO write pointer has a certain number of free spaces remaining. The trigger number can be set by the FIFO_A_FULL[3:0] register. The interrupt is cleared by reading the Interrupt Status 1 register (0x00).

PPG_RDY: New FIFO Data Ready

In SpO_2 and HR modes, this interrupt triggers when there is a new sample in the data FIFO. The interrupt is cleared by reading the Interrupt Status 1 register (0x00), or by reading the FIFO_DATA register.

ALC_OVF: Ambient Light Cancellation Overflow

This interrupt triggers when the ambient light cancellation function of the SpO_2/HR photodiode has reached its maximum limit, and therefore, ambient light is affecting the output of the ADC. The interrupt is cleared by reading the Interrupt Status 1 register (0x00).

PWR_RDY: Power Ready Flag

On power-up or after a brownout condition, when the supply voltage V_{DD} transitions from below the undervoltage lockout (UVLO) voltage to above the UVLO voltage, a power-ready interrupt is triggered to signal that the module is powered-up and ready to collect data.

DIE_TEMP_RDY: Internal Temperature Ready Flag

When an internal die temperature conversion is finished, this interrupt is triggered so the processor can read the temperature data registers. The interrupt is cleared by reading either the Interrupt Status 2 register (0x01) or the TFRAC register (0x20).

The interrupts are cleared whenever the interrupt status register is read, or when the register that triggered the interrupt is read. For example, if the SpO_2 sensor triggers an interrupt due to finishing a conversion, reading either the FIFO data register or the interrupt register clears the interrupt pin (which returns to its normal HIGH state). This also clears all the bits in the interrupt status register to zero.

Interrupt Enable (0x02-0x03)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
Interrupt Enable 1	A_ FULL_ EN	PPG_ RDY_EN	ALC_ OVF_EN						0x02	0X00	R/W
Interrupt Enable 2							DIE_TEMP_ RDY_EN		0x03	0x00	R/W

Each source of hardware interrupt, with the exception of power ready, can be disabled in a software register within the MAX30102 IC. The power-ready interrupt cannot be disabled because the digital state of the module is reset upon a brownout condition (low power supply voltage), and the default condition is that all the interrupts are disabled. Also, it is important for the system to know that a brownout condition has occurred, and the data within the module is reset as a result.

The unused bits should always be set to zero for normal operation.

FIFO (0x04–0x07)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
FIFO Write Pointer				FIFO_WR_PTR[4:0]					0x04	0x00	R/W
Over Flow Counter				OVF_COUNTER[4:0]					0x05	0x00	R/W
FIFO Read Pointer				FIFO_RD_PTR[4:0]					0x06	0x00	R/W
FIFO Data Register	FIFO_DATA[7:0]								0x07	0x00	R/W

FIFO Write Pointer

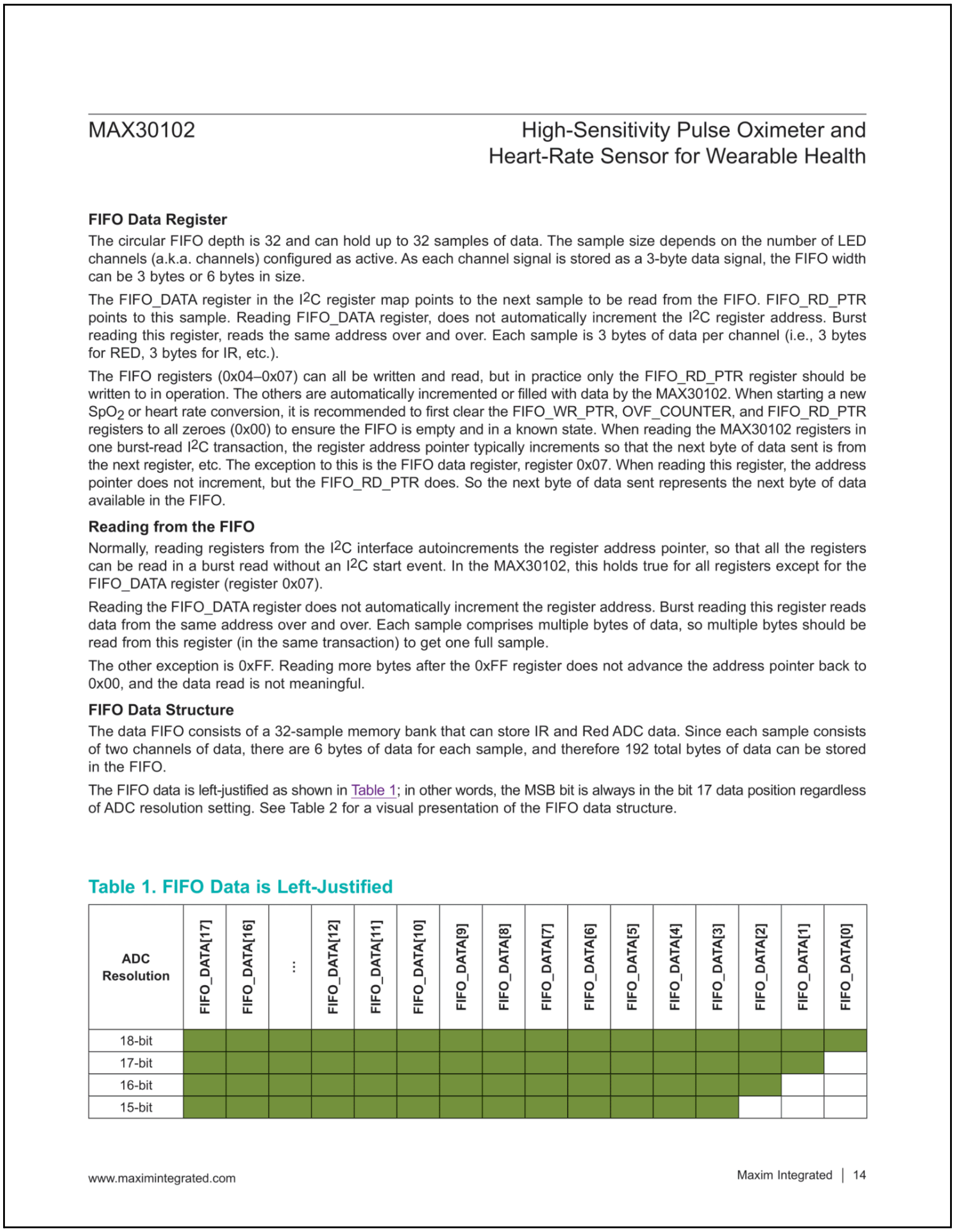
The FIFO Write Pointer points to the location where the MAX30102 writes the next sample. This pointer advances for each sample pushed on to the FIFO. It can also be changed through the I^2C interface when MODE[2:0] is 010, 011, or 111.

FIFO Overflow Counter

When the FIFO is full, samples are not pushed on to the FIFO, samples are lost. OVF_COUNTER counts the number of samples lost. It saturates at 0x1F. When a complete sample is "popped" (i.e., removal of old FIFO data and shifting the samples down) from the FIFO (when the read pointer advances), OVF_COUNTER is reset to zero.

FIFO Read Pointer

The FIFO Read Pointer points to the location from where the processor gets the next sample from the FIFO through the I^2C interface. This advances each time a sample is popped from the FIFO. The processor can also write to this pointer after reading the samples to allow rereading samples from the FIFO if there is a data communication error.



FIFO Data Register

The circular FIFO depth is 32 and can hold up to 32 samples of data. The sample size depends on the number of LED channels (a.k.a. channels) configured as active. As each channel signal is stored as a 3-byte data signal, the FIFO width can be 3 bytes or 6 bytes in size.

The FIFO_DATA register in the I^2C register map points to the next sample to be read from the FIFO. FIFO_RD_PTR points to this sample. Reading FIFO_DATA register, does not automatically increment the I^2C register address. Burst reading this register, reads the same address over and over. Each sample is 3 bytes of data per channel (i.e., 3 bytes for RED, 3 bytes for IR, etc.).

The FIFO registers (0x04–0x07) can all be written and read, but in practice only the FIFO_RD_PTR register should be written to in operation. The others are automatically incremented or filled with data by the MAX30102. When starting a new SpO_2 or heart rate conversion, it is recommended to first clear the FIFO_WR_PTR, OVF_COUNTER, and FIFO_RD_PTR registers to all zeroes (0x00) to ensure the FIFO is empty and in a known state. When reading the MAX30102 registers in one burst-read I^2C transaction, the register address pointer typically increments so that the next byte of data sent is from the next register, etc. The exception to this is the FIFO data register, register 0x07. When reading this register, the address pointer does not increment, but the FIFO_RD_PTR does. So the next byte of data sent represents the next byte of data available in the FIFO.

Reading from the FIFO

Normally, reading registers from the I^2C interface autoincrements the register address pointer, so that all the registers can be read in a burst read without an I^2C start event. In the MAX30102, this holds true for all registers except for the FIFO_DATA register (register 0x07).

Reading the FIFO_DATA register does not automatically increment the register address. Burst reading this register reads data from the same address over and over. Each sample comprises multiple bytes of data, so multiple bytes should be read from this register (in the same transaction) to get one full sample.

The other exception is 0xFF. Reading more bytes after the 0xFF register does not advance the address pointer back to 0x00, and the data read is not meaningful.

FIFO Data Structure

The data FIFO consists of a 32-sample memory bank that can store IR and Red ADC data. Since each sample consists of two channels of data, there are 6 bytes of data for each sample, and therefore 192 total bytes of data can be stored in the FIFO.

The FIFO data is left-justified as shown in Table 1; in other words, the MSB bit is always in the bit 17 data position regardless of ADC resolution setting. See Table 2 for a visual presentation of the FIFO data structure.

Table 1. FIFO Data is Left-Justified

ADC Resolution	FIFO_DATA[17]	FIFO_DATA[16]	...	FIFO_DATA[12]	FIFO_DATA[11]	FIFO_DATA[10]	FIFO_DATA[9]	FIFO_DATA[8]	FIFO_DATA[7]	FIFO_DATA[6]	FIFO_DATA[5]	FIFO_DATA[4]	FIFO_DATA[3]	FIFO_DATA[2]	FIFO_DATA[1]	FIFO_DATA[0]
18-bit	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
17-bit	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	
16-bit	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩		
15-bit	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩			

FIFO Data Contains 3 Bytes per Channel

The FIFO data is left-justified, meaning that the MSB is always in the same location regardless of the ADC resolution setting. FIFO DATA[18] – [23] are not used. Table 2 shows the structure of each triplet of bytes (containing the 18-bit ADC data output of each channel).

Each data sample in SpO_2 mode comprises two data triplets (3 bytes each), To read one sample, requires an I^2C read command for each byte. Thus, to read one sample in SpO_2 mode, requires 6 I^2C byte reads. The FIFO read pointer is automatically incremented after the first byte of each sample is read.

Write/Read Pointers

Write/Read pointers are used to control the flow of data in the FIFO. The write pointer increments every time a new sample is added to the FIFO. The read pointer is incremented every time a sample is read from the FIFO. To reread a sample from the FIFO, decrement its value by one and read the data register again.

The FIFO write/read pointers should be cleared (back to 0x00) upon entering SpO_2 mode or HR mode, so that there is no old data represented in the FIFO. The pointers are automatically cleared if V_{DD} is power-cycled or V_{DD} drops below its UVLO voltage.

BYTE 1	▨	▨	▨	▨	▨	▨	FIFO_ DATA[17]	FIFO_ DATA[16]
BYTE 2	FIFO_ DATA[15]	FIFO_ DATA[14]	FIFO_ DATA[13]	FIFO_ DATA[12]	FIFO_ DATA[11]	FIFO_ DATA[10]	FIFO_ DATA[9]	FIFO_ DATA[8]
BYTE 3	FIFO_ DATA[7]	FIFO_ DATA[6]	FIFO_ DATA[5]	FIFO_ DATA[4]	FIFO_ DATA[3]	FIFO_ DATA[2]	FIFO_ DATA[1]	FIFO_ DATA[0]

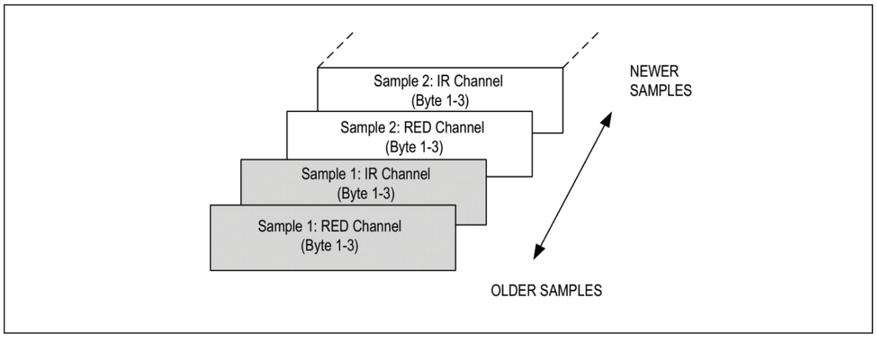


Figure 2. Graphical Representation of the FIFO Data Register. It shows IR and Red in SpO_2 Mode.

Pseudo-Code Example of Reading Data from FIFO

First transaction: Get the FIFO_WR_PTR:

```
START;
Send device address + write mode
Send address of FIFO_WR_PTR;
REPEATED_START;
Send device address + read mode
Read FIFO_WR_PTR;
STOP;
```

The central processor evaluates the number of samples to be read from the FIFO:

```
NUM_AVAILABLE_SAMPLES = FIFO_WR_PTR – FIFO_RD_PTR
(Note: pointer wrap around should be taken into account)
NUM_SAMPLES_TO_READ = < less than or equal to NUM_AVAILABLE_SAMPLES >
```

Second transaction: Read `NUM_SAMPLES_TO_READ` samples from the FIFO:

```
START;
Send device address + write mode
Send address of FIFO_DATA;
REPEATED_START;
Send device address + read mode
for (i = 0; i < NUM_SAMPLES_TO_READ; i++) {
Read FIFO_DATA;
Save LED1[23:16];
Read FIFO_DATA;
Save LED1[15:8];
Read FIFO_DATA;
Save LED1[7:0];
Read FIFO_DATA;
Save LED2[23:16];
Read FIFO_DATA;
Save LED2[15:8];
Read FIFO_DATA;
Save LED2[7:0];
Read FIFO_DATA;
}
STOP;
START;
Send device address + write mode
Send address of FIFO_RD_PTR;
Write FIFO_RD_PTR;
STOP;
```

Third transaction: Write to FIFO_RD_PTR register. If the second transaction was successful, FIFO_RD_PTR points to the next sample in the FIFO, and this third transaction is not necessary. Otherwise, the processor updates the FIFO_RD_PTR appropriately, so that the samples are reread.

FIFO Configuration (0x08)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
FIFO Configuration	SMP_AVE[2:0]			FIFO_ROLLOVER_EN	FIFO_A_FULL[3:0]				0x08	0x00	R/W

Bits 7:5: Sample Averaging (SMP_AVE)

To reduce the amount of data throughput, adjacent samples (in each individual channel) can be averaged and decimated on the chip by setting this register.

Table 3. Sample Averaging

SMP_AVE[2:0]	NO. OF SAMPLES AVERAGED PER FIFO SAMPLE
000	1 (no averaging)
001	2
010	4
011	8
100	16
101	32
110	32
111	32

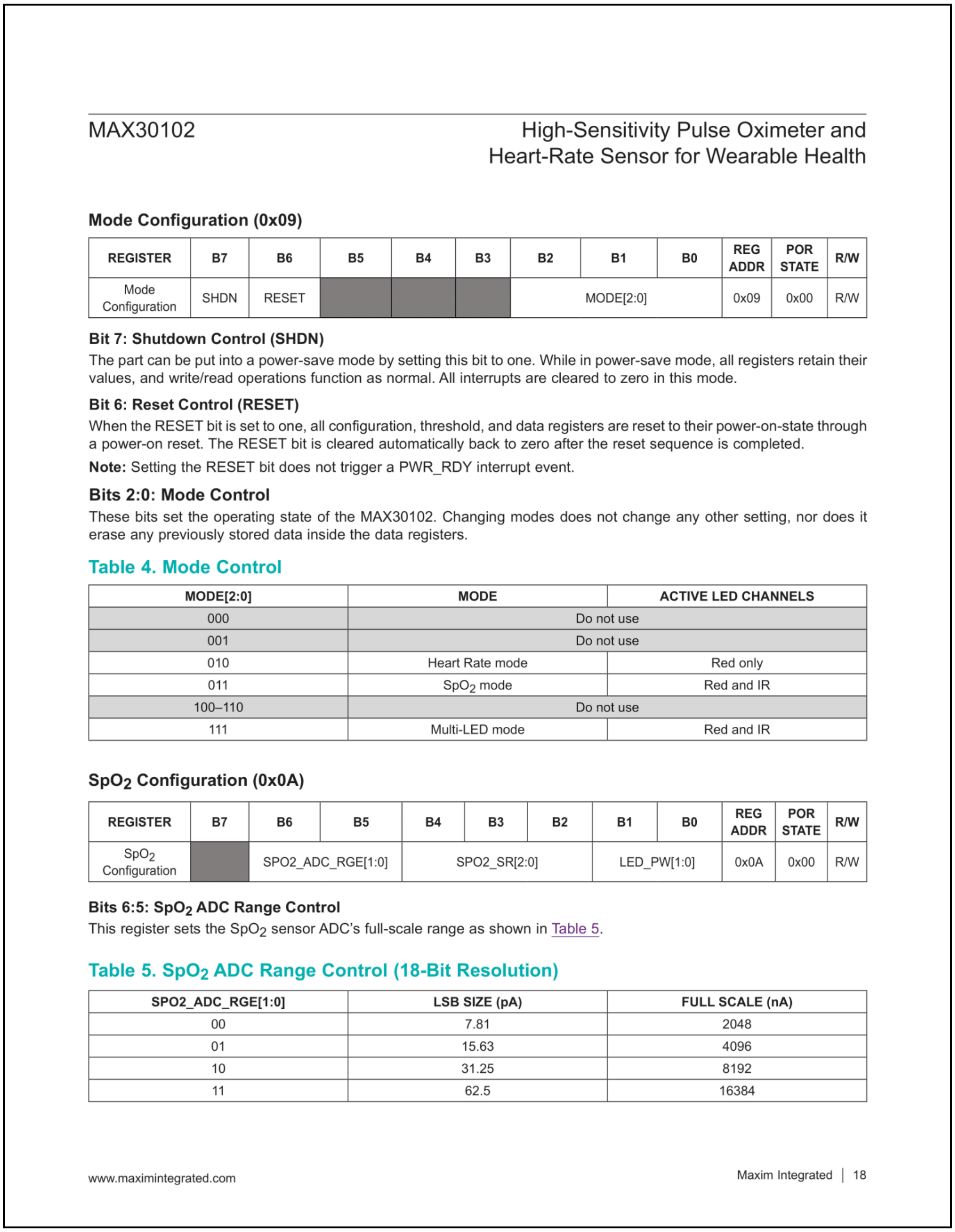
Bit 4: FIFO Rolls on Full (FIFO_ROLLOVER_EN)

This bit controls the behavior of the FIFO when the FIFO becomes completely filled with data. If FIFO_ROLLOVER_EN is set (1), the FIFO address rolls over to zero and the FIFO continues to fill with new data. If the bit is not set (0), then the FIFO is not updated until FIFO_DATA is read or the WRITE/READ pointer positions are changed.

Bits 3:0: FIFO Almost Full Value (FIFO_A_FULL)

This register sets the number of data samples (3 bytes/sample) remaining in the FIFO when the interrupt is issued. For example, if this field is set to 0x0, the interrupt is issued when there is 0 data samples remaining in the FIFO (all 32 FIFO words have unread data). Furthermore, if this field is set to 0xF, the interrupt is issued when 15 data samples are remaining in the FIFO (17 FIFO data samples have unread data).

FIFO_A_FULL[3:0]	EMPTY DATA SAMPLES IN FIFO WHEN INTERRUPT IS ISSUED	UNREAD DATA SAMPLES IN FIFO WHEN INTERRUPT IS ISSUED
0x0h	0	32
0x1h	1	31
0x2h	2	30
0x3h	3	29
…	…	…
0xFh	15	17



Mode Configuration (0x09)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
Mode Configuration	SHDN	RESET	▨	▨	▨	MODE[2:0]			0x09	0x00	R/W

Bit 7: Shutdown Control (SHDN)

The part can be put into a power-save mode by setting this bit to one. While in power-save mode, all registers retain their values, and write/read operations function as normal. All interrupts are cleared to zero in this mode.

Bit 6: Reset Control (RESET)

When the RESET bit is set to one, all configuration, threshold, and data registers are reset to their power-on-state through a power-on reset. The RESET bit is cleared automatically back to zero after the reset sequence is completed.

Note: Setting the RESET bit does not trigger a PWR_RDY interrupt event.

Bits 2:0: Mode Control

These bits set the operating state of the MAX30102. Changing modes does not change any other setting, nor does it erase any previously stored data inside the data registers.

Table 4. Mode Control

MODE[2:0]	MODE	ACTIVE LED CHANNELS
000	Do not use	
001	Do not use	
010	Heart Rate mode	Red only
011	SpO_2 mode	Red and IR
100–110	Do not use	
111	Multi-LED mode	Red and IR

SpO_2 Configuration (0x0A)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
SpO_2 Configuration	▨	SPO2_ADC_RGE[1:0]		SPO2_SR[2:0]			LED_PW[1:0]		0x0A	0x00	R/W

Bits 6:5: SpO_2 ADC Range Control

This register sets the SpO_2 sensor ADC's full-scale range as shown in Table 5.

Table 5. SpO_2 ADC Range Control (18-Bit Resolution)

SPO2_ADC_RGE[1:0]	LSB SIZE (pA)	FULL SCALE (nA)
00	7.81	2048
01	15.63	4096
10	31.25	8192
11	62.5	16384

Bits 4:2: SpO_2 Sample Rate Control

These bits define the effective sampling rate with one sample consisting of one IR pulse/conversion and one Red pulse/conversion.

The sample rate and pulse width are related in that the sample rate sets an upper bound on the pulse width time. If the user selects a sample rate that is too high for the selected LED_PW setting, the highest possible sample rate is programmed instead into the register.

Table 6. SpO_2 Sample Rate Control

SPO2_SR[2:0]	SAMPLES PER SECOND
000	50
001	100
010	200
011	400
100	800
101	1000
110	1600
111	3200

See Table 11 and Table 12 for Pulse Width vs. Sample Rate information.

Bits 1:0: LED Pulse Width Control and ADC Resolution

These bits set the LED pulse width (the IR and Red have the same pulse width), and therefore, indirectly sets the integration time of the ADC in each sample. The ADC resolution is directly related to the integration time.

Table 7. LED Pulse Width Control

LED_PW[1:0]	PULSE WIDTH (µs)	ADC RESOLUTION (bits)
00	69 (68.95)	15
01	118 (117.78)	16
10	215 (215.44)	17
11	411 (410.75)	18

LED Pulse Amplitude (0x0C–0x0D)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
LED Pulse Amplitude	LED1_PA[7:0]								0x0C	0x00	R/W
	LED2_PA[7:0]								0x0D	0x00	R/W

These bits set the current level of each LED as shown in Table 8.

Table 8. LED Current Control

LEDx_PA [7:0], RED_PA[7:0], or IR_PA[7:0]	TYPICAL LED CURRENT (mA)*
0x00h	0.0
0x01h	0.2
0x02h	0.4
…	…
0x0Fh	3.0
…	…
0x1Fh	6.2
…	…
0x3Fh	12.6
…	…
0x7Fh	25.4
…	…
0xFFh	51.0

**Actual measured LED current for each part can vary widely due to the trimming methodology.*

Multi-LED Mode Control Registers (0x11–0x12)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
Multi-LED Mode Control Registers		SLOT2[2:0]				SLOT1[2:0]			0x11	0x00	R/W
		SLOT4[2:0]				SLOT3[2:0]			0x12	0x00	R/W

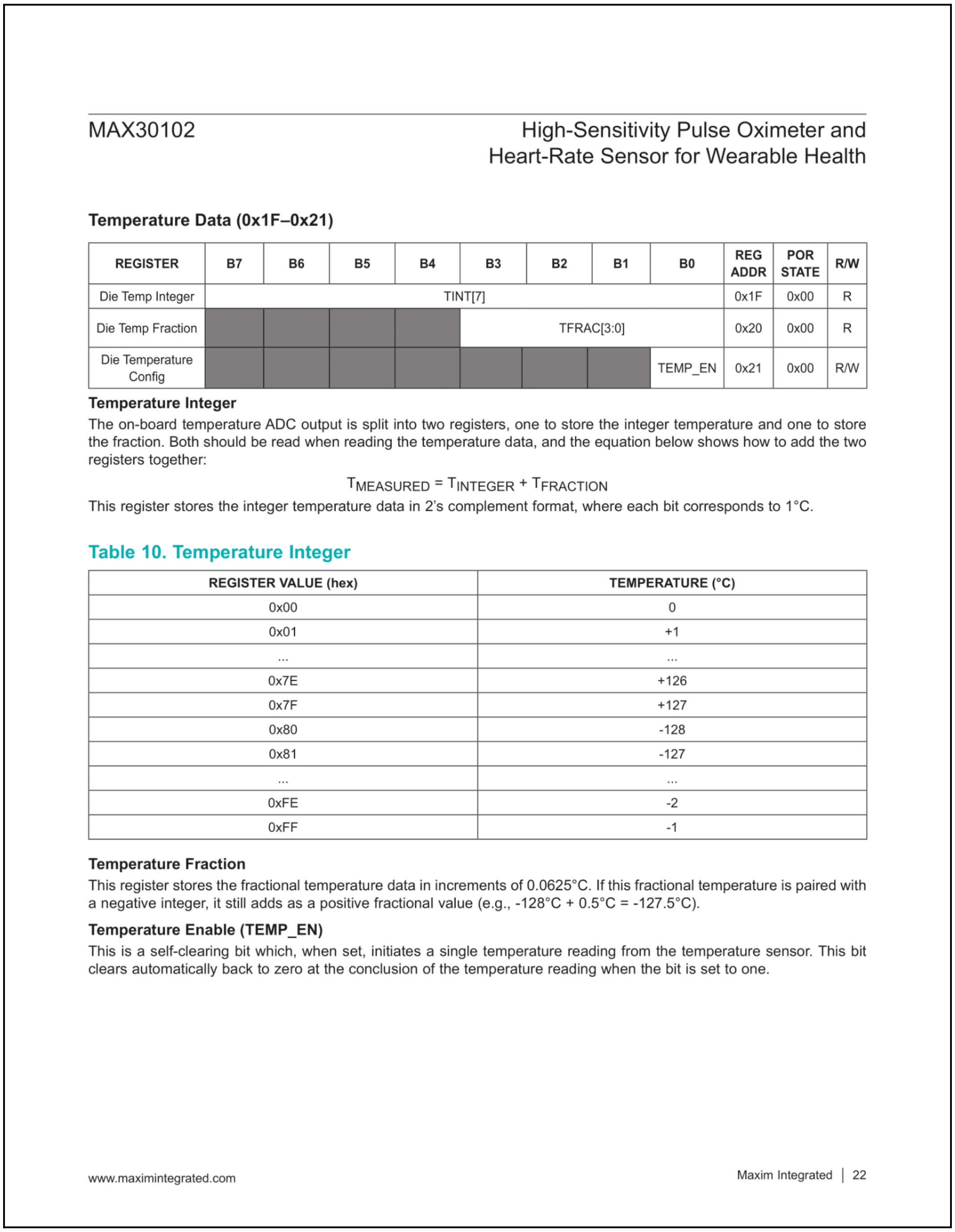
In multi-LED mode, each sample is split into up to four time slots, SLOT1 through SLOT4. These control registers determine which LED is active in each time slot, making for a very flexible configuration.

Table 9. Multi-LED Mode Control Registers

SLOTx[2:0] Setting	WHICH LED IS ACTIVE	LED PULSE AMPLITUDE SETTING
000	None (time slot is disabled)	N/A (Off)
001	LED1 (Red)	LED1_PA[7:0]
010	LED2 (IR)	LED2_PA[7:0]
011	None	N/A (Off)
100	None	N/A (Off)
101	Reserved	Reserved
110	Reserved	Reserved
111	Reserved	Reserved

Each slot generates a 3-byte output into the FIFO. One sample comprises all active slots, for example if SLOT1 and SLOT2 are non-zero, then one sample is 2 x 3 = 6 bytes.

The slots should be enabled in order (i.e., SLOT1 should not be disabled if SLOT2 is enabled).



Temperature Data (0x1F–0x21)

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0	REG ADDR	POR STATE	R/W
Die Temp Integer	TINT[7]								0x1F	0x00	R
Die Temp Fraction	▨	▨	▨	▨	TFRAC[3:0]				0x20	0x00	R
Die Temperature Config	▨	▨	▨	▨	▨	▨	▨	TEMP_EN	0x21	0x00	R/W

Temperature Integer

The on-board temperature ADC output is split into two registers, one to store the integer temperature and one to store the fraction. Both should be read when reading the temperature data, and the equation below shows how to add the two registers together:

$$T_{MEASURED} = T_{INTEGER} + T_{FRACTION}$$

This register stores the integer temperature data in 2's complement format, where each bit corresponds to 1°C.

Table 10. Temperature Integer

REGISTER VALUE (hex)	TEMPERATURE (°C)
0x00	0
0x01	+1
...	...
0x7E	+126
0x7F	+127
0x80	-128
0x81	-127
...	...
0xFE	-2
0xFF	-1

Temperature Fraction

This register stores the fractional temperature data in increments of 0.0625°C. If this fractional temperature is paired with a negative integer, it still adds as a positive fractional value (e.g., -128°C + 0.5°C = -127.5°C).

Temperature Enable (TEMP_EN)

This is a self-clearing bit which, when set, initiates a single temperature reading from the temperature sensor. This bit clears automatically back to zero at the conclusion of the temperature reading when the bit is set to one.

Applications Information

Sample Rate and Performance

The maximum sample rate for the ADC depends on the selected pulse width, which in turn, determines the ADC resolution. For instance, if the pulse width is set to 69µs then the ADC resolution is 15 bits, and all sample rates are selectable. However, if the pulse width is set to 411µs, then the samples rates are limited. The allowed sample rates for both SpO_2 and HR Modes are summarized in the Table 11 and Table 12.

Power Considerations

The LED waveforms and their implication for power supply design are discussed in this section.

The LEDs in the MAX30102 are pulsed with a low duty cycle for power savings, and the pulsed currents can cause ripples in the V_{LED+} power supply. To ensure these pulses do not translate into optical noise at the LED outputs, the power supply must be designed to handle these. Ensure that the resistance and inductance from the power supply (battery, DC/DC converter, or LDO) to the pin is much smaller than 1Ω, and that there is at least 1µF of power supply bypass capacitance to a good ground plane. The capacitance should be located as close as physically possible to the IC.

Table 11. SpO_2 Mode (Allowed Settings)

SAMPLES PER SECOND	PULSE WIDTH (µs)			
	69	118	215	411
50	O	O	O	O
100	O	O	O	O
200	O	O	O	O
400	O	O	O	O
800	O	O	O	▨
1000	O	O	▨	▨
1600	O	▨	▨	▨
3200	▨	▨	▨	▨
Resolution (bits)	15	16	17	18

Table 12. HR Mode (Allowed Settings)

SAMPLES PER SECOND	PULSE WIDTH (µs)			
	69	118	215	411
50	O	O	O	O
100	O	O	O	O
200	O	O	O	O
400	O	O	O	O
800	O	O	O	O
1000	O	O	O	O
1600	O	O	O	▨
3200	O	▨	▨	▨
Resolution (bits)	15	16	17	18

In the Heart Rate mode, only the Red LED is used to capture optical data and determine the user's heart rate and/or photoplethysmogram (PPG).

SpO_2 Temperature Compensation

The MAX30102 has an accurate on-board temperature sensor that digitizes the IC's internal temperature upon command from the I^2C master. The temperature has an effect on the wavelength of the red and IR LEDs. While the device output data is relatively insensitive to the wavelength of the IR LED, the red LED's wavelength is critical to correct interpretation of the data.

Table 13 shows the correlation of red LED wavelength versus the temperature of the LED. Since the LED die heats up with a very short thermal time constant (tens of microseconds), the LED wavelength should be calculated according to the current level of the LED and the temperature of the IC. Use Table 13 to estimate the temperature.

Red LED Current Settings vs. LED Temperature Rise

Add the temperature rise to the module temperature reading to estimate the LED temperature and output wavelength. The LED temperature estimate is valid even with very short pulse widths, due to the fast thermal time constant of the LED.

Interrupt Pin Functionality

The active-low interrupt pin pulls low when an interrupt is triggered. The pin is open-drain, which means it normally requires a pullup resistor or current source to an external voltage supply (up to +5V from GND). The interrupt pin is not designed to sink large currents, so the pullup resistor value should be large, such as 4.7kΩ.

Table 13. RED LED Current Settings vs. LED Temperature Rise

RED LED CURRENT SETTING	RED LED DUTY CYCLE (% OF LED PULSE WIDTH TO SAMPLE TIME)	ESTIMATED TEMPERATURE RISE (ADD TO TEMP SENSOR MEASUREMENT) (°C)
00000001 (0.2mA)	8	0.1
11111010 (50mA)	8	2
00000001 (0.2mA)	16	0.3
11111010 (50mA)	16	4
00000001 (0.2mA)	32	0.6
11111010 (50mA)	32	8

Timing for Measurements and Data Collection

Slot Timing in Multi-LED Modes

The MAX30102 can support two LED channels of sequential processing (Red and IR). Table 14 below displays the four possible channel slot times associated with each pulse width setting. Figure 3 shows an example of channel slot timing for a SpO_2 mode application with a 1kHz sample rate.

Table 14. Slot Timing

PULSE-WIDTH SETTING (µs)	CHANNEL SLOT TIMING (TIMING PERIOD BETWEEN PULSES) (µs)	CHANNEL-CHANNEL TIMING (RISING EDGE-TO-RISING EDGE) (µs)
69	358	427
118	407	525
215	505	720
411	696	1107

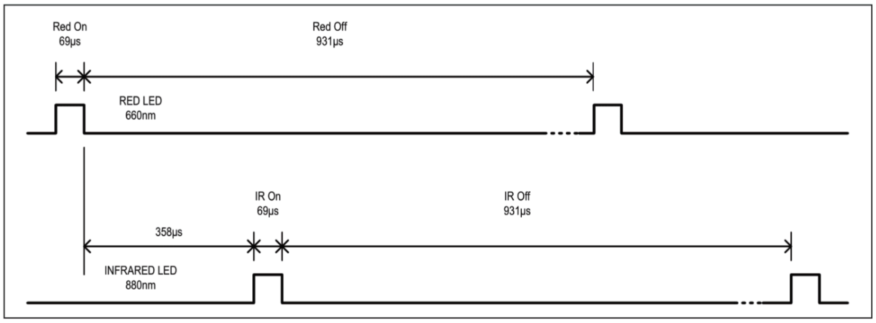


Figure 3. Channel Slot Timing for the SpO_2 Mode with a 1kHz Sample Rate

Timing in SpO_2 Mode

The internal FIFO stores up to 32 samples, so that the system processor does not need to read the data after every sample. The temperature does not need to be sampled very often–once a second or every few seconds should be sufficient.

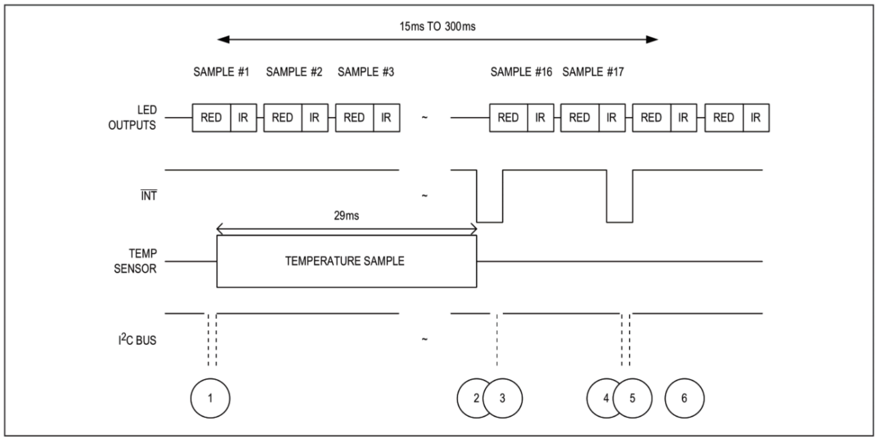


Figure 4. Timing for Data Acquisition and Communication When in SpO_2 Mode

Table 15. Events Sequence for Figure 4 in SpO_2 Mode

EVENT	DESCRIPTION	COMMENTS
1	Enter into SpO_2 Mode. Initiate a Temperature measurement.	I^2C Write Command sets MODE[2:0] = 0x03 and A_FULL_EN. Then to enable and initiate a single temperature measurement, set TEMP_EN and DIE_TEMP_RDY_EN.
2	Temperature Measurement Complete, Interrupt Generated	DIE_TEMP_RDY interrupt triggers, alerting the central processor to read the data.
3	Temp Data is Read, Interrupt Cleared	
4	FIFO is Almost Full, Interrupt Generated	Interrupt is generated when the FIFO almost full threshold is reached.
5	FIFO Data is Read, Interrupt Cleared	
6	Next Sample is Stored	New Sample is stored at the new read pointer location. Effectively, it is now the first sample in the FIFO.

Timing in HR Mode

The internal FIFO stores up to 32 samples, so that the system processor does not need to read the data after every sample. In HR mode (Figure 5), unlike in SpO_2 mode, temperature information is not necessary to interpret the data. The user can select either the red LED or the infrared LED channel for heart rate measurements.

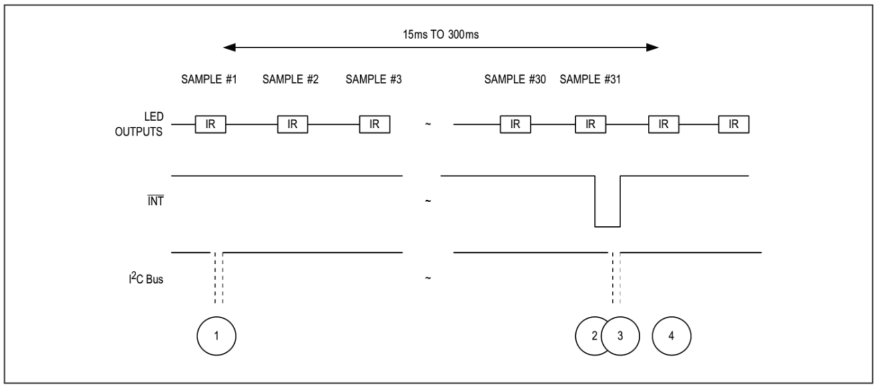


Figure 5. Timing for Data Acquisition and Communication When in HR Mode

Table 16. Events Sequence for Figure 5 in HR Mode

EVENT	DESCRIPTION	COMMENTS
1	Enter into Mode	I^2C Write Command sets MODE[2:0] = 0x02. Mask the A_FULL_EN Interrupt.
2	FIFO is Almost Full, Interrupt Generated	Interrupt is generated when the FIFO has only one empty space left.
3	FIFO Data is Read, Interrupt Cleared	
4	Next Sample is Stored	New sample is stored at the new read pointer location. Effectively, it is now the first sample in the FIFO.

Power Sequencing and Requirements

Power-Up Sequencing

Figure 6. shows the recommended power-up sequence for the MAX30102.

It is recommended to power the V_{DD} supply first, before the LED power supplies (V_{LED+}). The interrupt and I^{2}C pins can be pulled up to an external voltage even when the power supplies are not powered up.

After the power is established, an interrupt occurs to alert the system that the MAX30102 is ready for operation. Reading the I^{2}C interrupt register clears the interrupt, as shown in Figure 6.

Power-Down Sequencing

The MAX30102 is designed to be tolerant of any power supply sequencing on power-down.

I^{2}C Interface

The MAX30102 features an I^{2}C/SMBus-compatible, 2-wire serial interface consisting of a serial data line (SDA) and a serial clock line (SCL). SDA and SCL facilitate communication between the MAX30102 and the master at clock rates up to 400kHz. Figure 1 shows the 2-wire interface timing diagram. The master generates SCL and initiates data transfer on the bus. The master device writes data to the MAX30102 by transmitting the proper slave address followed by data. Each transmit sequence is framed by a START (S) or REPEATED START (Sr) condition and a STOP (P) condition. Each word transmitted to the MAX30102 is 8 bits long and is followed by an acknowledge clock pulse. A master reading data from the MAX30102 transmits the proper slave address followed by a series of nine SCL pulses.

The MAX30102 transmits data on SDA in sync with the master-generated SCL pulses. The master acknowledges receipt of each byte of data. Each read sequence is framed by a START (S) or REPEATED START (Sr) condition, a not acknowledge, and a STOP (P) condition. SDA operates as both an input and an open-drain output. A pullup resistor, typically greater than 500Ω, is required on SDA. SCL operates only as an input. A pullup resistor, typically greater than 500Ω, is required on SCL if there are multiple masters on the bus, or if the single master has an open-drain SCL output. Series resistors in line with SDA and SCL are optional. Series resistors protect the digital inputs of the MAX30102 from high voltage spikes on the bus lines and minimize crosstalk and undershoot of the bus signals.

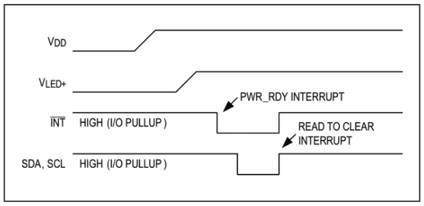


Figure 6. Power-Up Sequence of the Power Supply Rails

Bit Transfer

One data bit is transferred during each SCL cycle. The data on SDA must remain stable during the high period of the SCL pulse. Changes in SDA while SCL is high are control signals. See the *START and STOP Conditions* section.

START and STOP Conditions

SDA and SCL idle high when the bus is not in use. A master initiates communication by issuing a START condition. A START condition is a high-to-low transition on SDA with SCL high. A STOP condition is a low-to-high transition on SDA while SCL is high (Figure 7). A START condition from the master signals the beginning of a transmission to the device. The master terminates transmission, and frees the bus, by issuing a STOP condition. The bus remains active if a REPEATED START condition is generated instead of a STOP condition.

Early STOP Conditions

The MAX30102 recognizes a STOP condition at any point during data transmission except if the STOP condition occurs in the same high pulse as a START condition. For proper operation, do not send a STOP condition during the same SCL high pulse as the START condition.

Slave Address

A bus master initiates communication with a slave device by issuing a START condition followed by the 7-bit slave ID. When idle, the MAX30102 waits for a START condition followed by its slave ID. The serial interface compares each slave ID bit by bit, allowing the interface to power down and disconnect from SCL immediately if an incorrect slave ID is detected. After recognizing a START condition followed by the correct slave ID, the MAX30102 is programmed to accept or send data. The LSB of the slave ID word is the read/write ($R/\overline{W}$) bit. R/W indicates whether the master is writing to or reading data from the MAX30102 ($R/\overline{W}$ = 0 selects a write condition, $R/\overline{W}$ = 1 selects a read condition).

After receiving the proper slave ID, the MAX30102 issues an ACK by pulling SDA low for one clock cycle.

The MAX30102 slave ID consists of seven fixed bits, B7–B1 (set to 0b1010111). The most significant slave ID bit (B7) is transmitted first, followed by the remaining bits. Table 17 shows the possible slave IDs of the device.

Acknowledge

The acknowledge bit (ACK) is a clocked 9th bit that the MAX30102 uses to handshake receipt each byte of data when in write mode (Figure 8). The MAX30102 pulls down SDA during the entire master-generated 9th clock pulse if the previous byte is successfully received. Monitoring ACK allows for detection of unsuccessful data transfers. An unsuccessful data transfer occurs if a receiving device is busy or if a system fault has occurred. In the event of an unsuccessful data transfer, the bus master retries communication. The master pulls down SDA during the 9th clock cycle to acknowledge receipt of data when the MAX30102 is in read mode. An acknowledge is sent by the master after each read byte to allow data transfer to continue. A not-acknowledge is sent when the master reads the final byte of data from the MAX30102, followed by a STOP condition.

Write Data Format

For the write operation, send the slave ID as the first byte followed by the register address byte and then one or more data bytes. The register address pointer increments automatically after each byte of data received, so for example the entire register bank can be written by at one time. Terminate the data transfer with a STOP condition. The write operation is shown in Figure 9.

The internal register address pointer increments automatically, so writing additional data bytes fill the data registers in order.

Table 17. Slave ID Description

B7	B6	B5	B4	B3	B2	B1	B0	WRITE ADDRESS	READ ADDRESS
1	0	1	0	1	1	1	$R/\overline{W}$	0xAE	0xAF

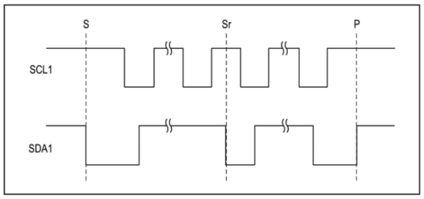


Figure 7. START, STOP, and REPEATED START Conditions

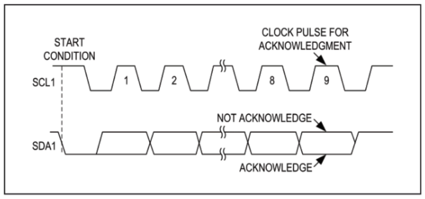


Figure 8. Acknowledge

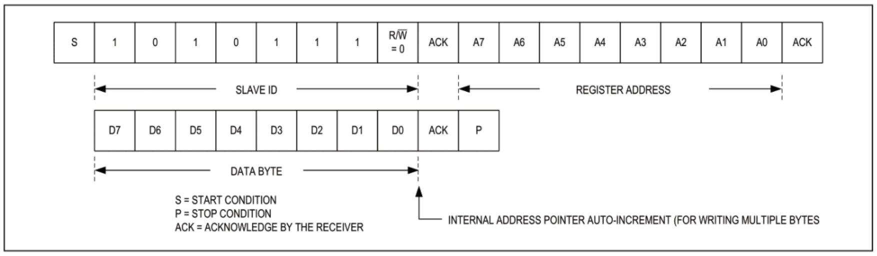


Figure 9. Writing One Data Byte to the MAX30102

Read Data Format

For the read operation, two I2C operations must be performed. First, the slave ID byte is sent followed by the I2C register that you wish to read. Then a REPEAT START (Sr) condition is sent, followed by the read slave ID. The MAX30102 then begins sending data beginning with the register selected in the first operation. The read pointer increments automatically, so the device continues sending data from additional registers in sequential order until a STOP (P) condition is received. The exception to this is the FIFO_DATA register, at which the read pointer no longer increments when reading additional bytes. To read the next register after FIFO_DATA, an I2C write command is necessary to change the location of the read pointer.

Figure 10 and Figure 11 show the process of reading one byte and multiple bytes of data.

An initial write operation is required to send the read register address.

Data is sent from registers in sequential order, starting from the register selected in the initial I2C write operation. If the FIFO_DATA register is read, the read pointer will not automatically increment, and subsequent bytes of data will contain the contents of the FIFO.

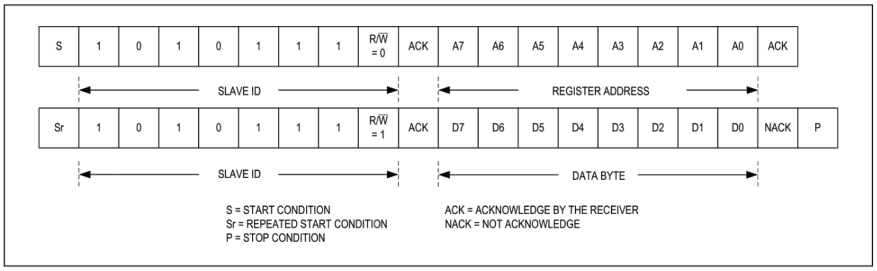


Figure 10. Reading One Byte of Data from MAX30102

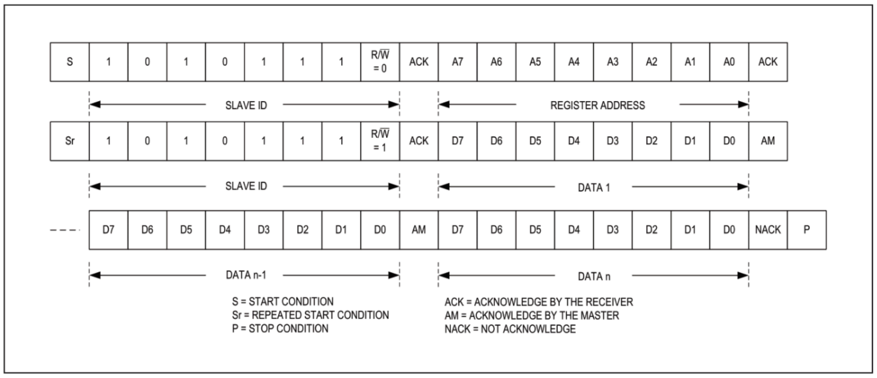
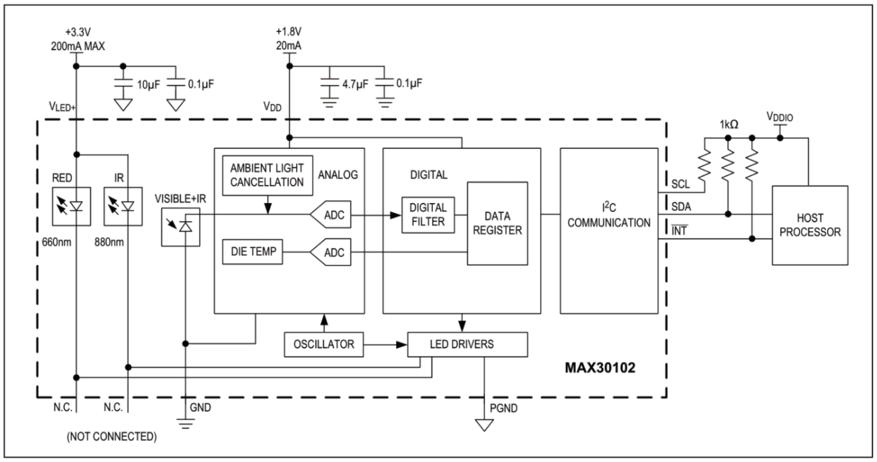


Figure 11. Reading Multiple Bytes of Data from the MAX30102

Typical Application Circuit



Ordering Information

PART	TEMP RANGE	PIN-PACKAGE
MAX30102EFD+T	-40°C to +85°C	14-Lead OESIP (0.8mm Pin Pitch)

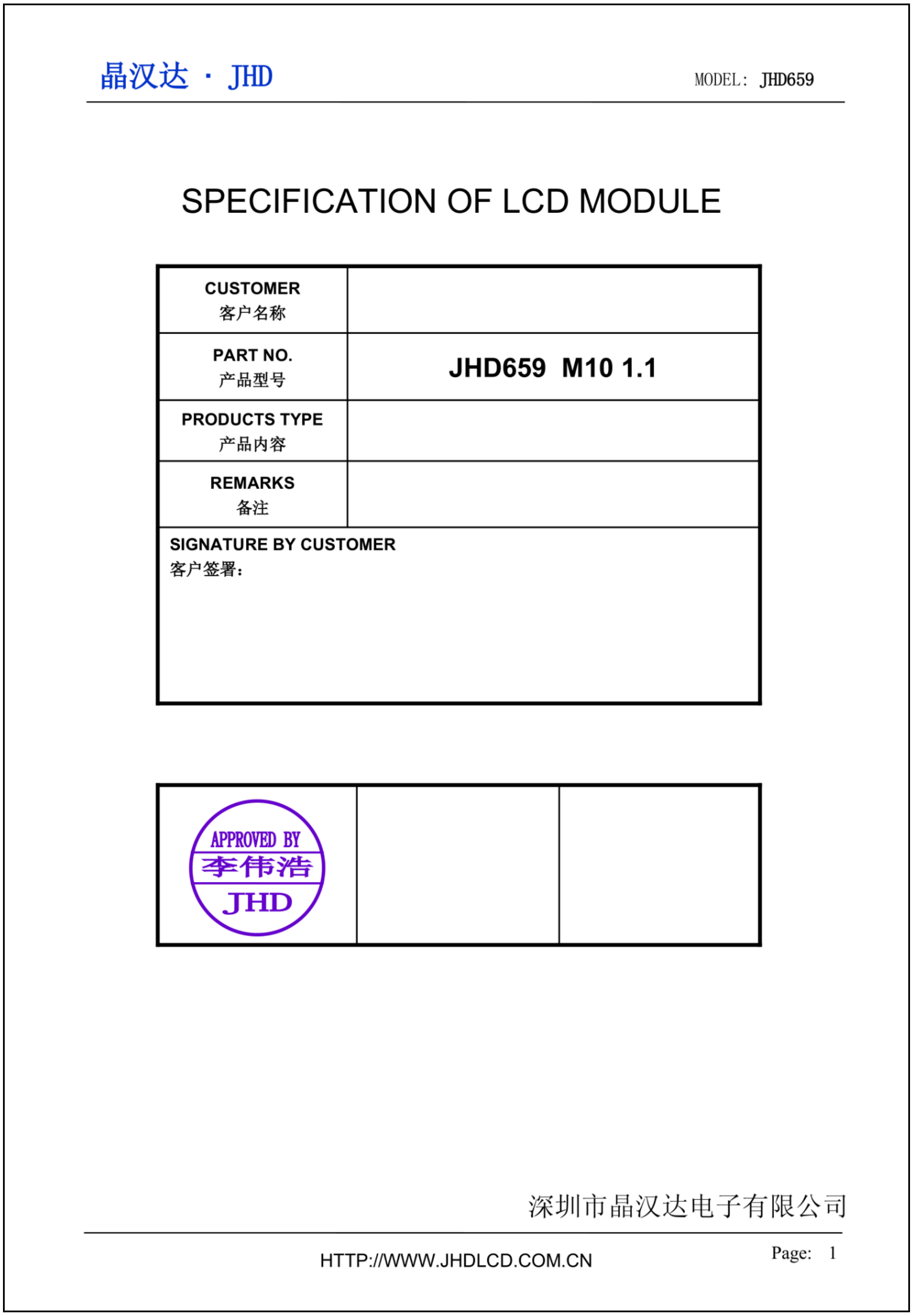
+Denotes lead(Pb)-free/RoHS-compliant package.
T = Tape and reel.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	9/15	Initial release	—
1	10/18	Updated the *General Description, Applications, Absolute Maximum Ratings, Electrical Characteristics, Pin Description, Timing in SpO*$_2$ *Mode, Power-Up Sequencing* sections; updated the *System Diagram, Pin Configuration, and Functional Diagram*; updated the *Register Map*, Interrupt Status (0x00–0x01), Interrupt Enable (0x02–0x03), FIFO (0x04–0x07), LED Pulse Amplitude (0x0C–0x0D), Table 8, Multi-LED Mode Control Registers (0x11–0x12), Table 9, Temperature Data (0x1F–0x21), Table 13, Table 15, Table 16; replaced the *Typical Application Circuit*; removed the *Proximity Function* section and the Proximity Mode Interrupt Threshold (0x30) register	1–5, 8–14, 18 20–24, 26–28, 31

For pricing, delivery, and ordering information, please visit Maxim Integrated's online storefront at https://www.maximintegrated.com/en/storefront/storefront.html.

16x2 LCD Display



晶汉达 · JHD

MODEL: JHD659

SPECIFICATION OF LCD MODULE

CUSTOMER 客户名称	
PART NO. 产品型号	**JHD659 M10 1.1**
PRODUCTS TYPE 产品内容	
REMARKS 备注	

SIGNATURE BY CUSTOMER

客户签署:

APPROVED BY 李伟浩 JHD		

深圳市晶汉达电子有限公司

HTTP://WWW.JHDLCD.COM.CN

Page: 1

LCM System

1 LCD Type

☒ S - STN ☐ F - FSTN ☐ D - DFSTN

2 Viewing Angle

☒ D - Lower 6:00 ☐ U - Upper 12:00 ☐ O - Others

3 Display Mode

☒ Yellow Green positive ☐ Blue Negative ☐ Grey positive

☐ FSTN positive ☐ FSTN negative

4 Polarizer Mode

☐ Reflective ☐ Transflective ☒ Transmissive

5 Connector

☐ Pin ☐ Heat sealed ☒ Zebra

6 Thickness of Glass

☒ 1.1mm ☐ 0.4mm

☐ 0.55mm ☐ 0.7mm

7 Backlight Mode:

☒ LED ☐ CCFL

8 Backlight Color

☐ Blue ☐ Amber ☒ Yellow Green

☐ Red ☐ White ☐ Without backlight

9 Temperature Grade

☒ Normal temperature ☐ Wide temperature ☐ Super wide temperature

10 CG-ROM

☒ 01 for English + Japanese language

•REVISION RECORD

REV. NO.	REV. DATE	DESCRIPTION OF REVISION	PAGE	REMARK
1.0	10/12/03	INITIAL RELEASE	ALL	
1.1	10/31/07	1、Change： Specification Edition. 2、Modify：OUTLINE DRAWING. 3、JHD659M10	ALL 5	

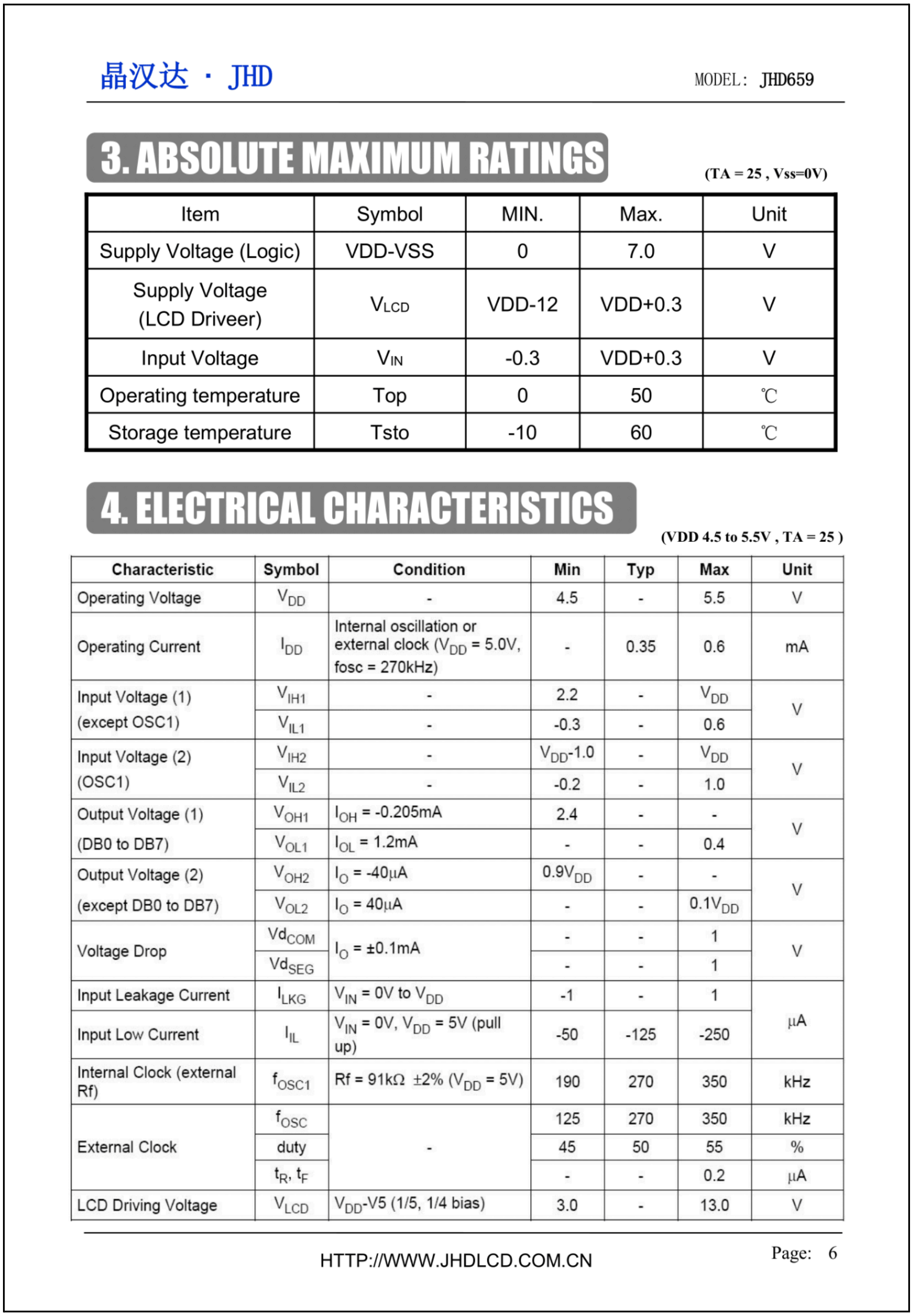
CONTENTS

1. FEATURES

•Display construction …………	16 Characters * 2 Lines
•Display mode ………………………	STN(Y/G)
•Display type ………………………	Positive Transmissive
•Backlight …………………………	LED/5.0V(Y/G)
•Viewing direction …………………	6 o' clock
•Operating temperature …………	0 to 50℃
•Storage temperature ……………	-10 to 60℃
•Controller …………………………	SPLC780D or Equivalence
•Driving voltage …………………	Single power
•Driving method …………………	1/16 duty, 1/5 bias
•Type ………………………………	COB (Chip On Board)
•Number of data line ……………	6800 4/8-bit parallel
•Connector …………………………	PIN

2. MECHANICAL DATA

ITEM		WIDTH	HEIGHT	THICKNESS	UNIT
Module size		80.0	36.0	13.5(MAX)	mm
Viewing area		64.5	14.5	-	mm
character	Construction	5*7			dots
	Size	2.95	4.35	-	mm
	Pitch	3.65	5.05	-	mm
Dot	Size	0.55	0.50	-	mm
	Pitch	0.60	0.55	-	mm
Diameter of mounting hole		Φ2.9			mm
Weight		About 50			g



3. ABSOLUTE MAXIMUM RATINGS

(TA = 25 , Vss=0V)

Item	Symbol	MIN.	Max.	Unit
Supply Voltage (Logic)	VDD-VSS	0	7.0	V
Supply Voltage (LCD Driveer)	V_{LCD}	VDD-12	VDD+0.3	V
Input Voltage	V_{IN}	-0.3	VDD+0.3	V
Operating temperature	Top	0	50	℃
Storage temperature	Tsto	-10	60	℃

4. ELECTRICAL CHARACTERISTICS

(VDD 4.5 to 5.5V , TA = 25)

Characteristic	Symbol	Condition	Min	Typ	Max	Unit
Operating Voltage	V_{DD}	-	4.5	-	5.5	V
Operating Current	I_{DD}	Internal oscillation or external clock (V_{DD} = 5.0V, fosc = 270kHz)	-	0.35	0.6	mA
Input Voltage (1) (except OSC1)	V_{IH1}	-	2.2	-	V_{DD}	V
	V_{IL1}	-	-0.3	-	0.6	
Input Voltage (2) (OSC1)	V_{IH2}	-	V_{DD}-1.0	-	V_{DD}	V
	V_{IL2}	-	-0.2	-	1.0	
Output Voltage (1) (DB0 to DB7)	V_{OH1}	I_{OH} = -0.205mA	2.4	-	-	V
	V_{OL1}	I_{OL} = 1.2mA	-	-	0.4	
Output Voltage (2) (except DB0 to DB7)	V_{OH2}	I_O = -40µA	0.9V_{DD}	-	-	V
	V_{OL2}	I_O = 40µA	-	-	0.1V_{DD}	
Voltage Drop	Vd_{COM}	I_O = ±0.1mA	-	-	1	V
	Vd_{SEG}		-	-	1	
Input Leakage Current	I_{LKG}	V_{IN} = 0V to V_{DD}	-1	-	1	µA
Input Low Current	I_{IL}	V_{IN} = 0V, V_{DD} = 5V (pull up)	-50	-125	-250	
Internal Clock (external Rf)	f_{OSC1}	Rf = 91kΩ ±2% (V_{DD} = 5V)	190	270	350	kHz
External Clock	f_{OSC}	-	125	270	350	kHz
	duty		45	50	55	%
	t_R, t_F		-	-	0.2	µA
LCD Driving Voltage	V_{LCD}	V_{DD}-V5 (1/5, 1/4 bias)	3.0	-	13.0	V

4.1 LED ELECTRICAL/OPTLCAL CHARACTERISTICS

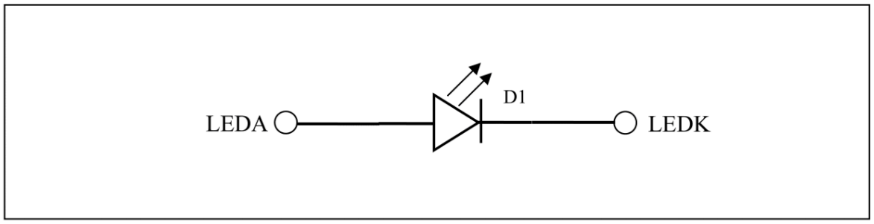
Item	Symbol	min	typ	max	Unit	Condition
Forward Voltage	Vf	–	5.0	5.2	V	If=20mA
Reverse Current	Ir	–	20	–	uA	Vr=5V
Dominant wave length	λp	565	-	575	nm	If=20mA
Spectral Line Half width	Δλ	–	30	–	nm	If=20mA
Luminance	Lv	–	60	–	cd/m²	If=20mA

4.2LED ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Condition	Rating	Unit
Reverse Voltage	Vr	Ta=25℃	5	V
Absolute maximum forward current	Ifm	Ta=25℃	25	mA
Power description	pd	Ta=25℃	125	mW

4.2.1 LED ARRAY BLOCK DIAGRAM

(LED DICE 1 dices)



4.2.2 LED POWER SOURCE

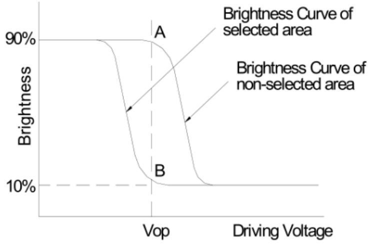
	Option	Power source	Jumper setting
LED	A	15A/16K	R7=110Ω

5. ELECTRO-OPTICAL CHARACTERISTICS

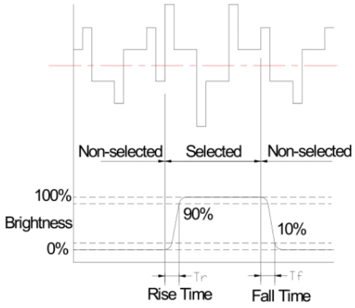
ITEM	SYMBOL	CONDITION	MIN.	TYP.	MAX.	UNIT	NOTE
Contrast ratio	K	Φ=0⁰	1.4	4	–	–	1
Response time (rise)	Tr	Φ=0⁰ θ=0⁰	–	130	–	ms	2
Response time (fall)	Tf	Φ=0⁰ θ=0⁰		130	–	ms	2
Viewing angle	Φ	K ≥1.4	-30 -- +30			deg.	3
	θ		-40 -- +15				

Note 1: Definition of Contrast Ratio "K"

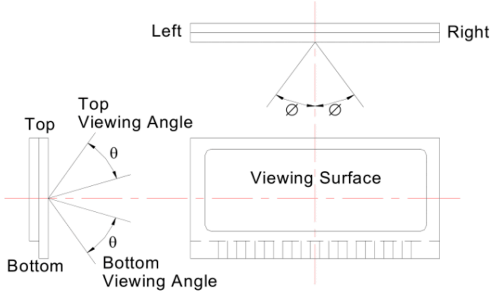
K= Brightness of non-selected segment(A) / Brightness of selected segment(B)



Note 2: Definition of Optical Response Time

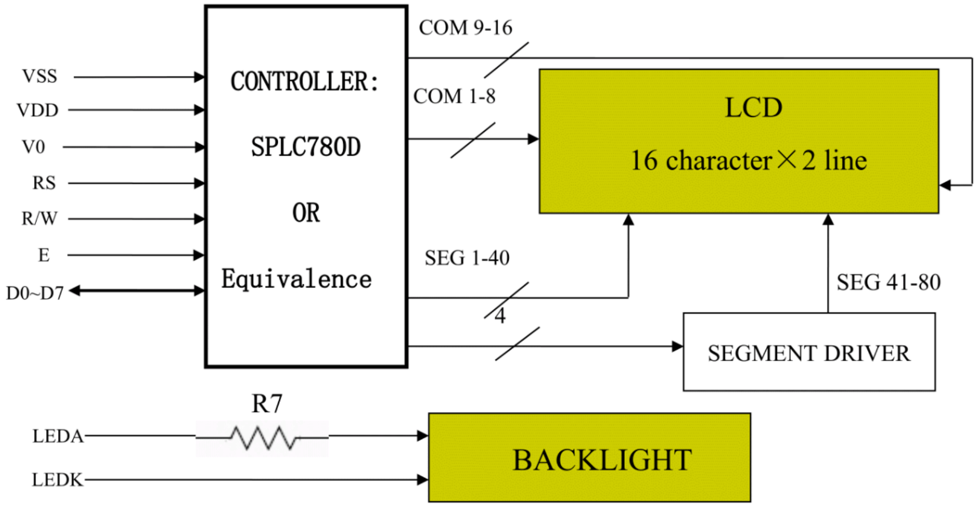


Note 3: Definition of Viewing Angle

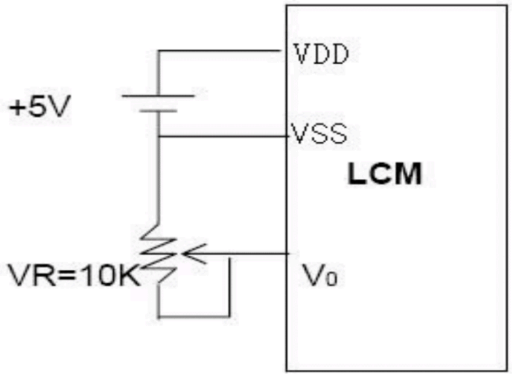


Please select either top or bottom viewing angle

6. BLOCK DIAGRAM

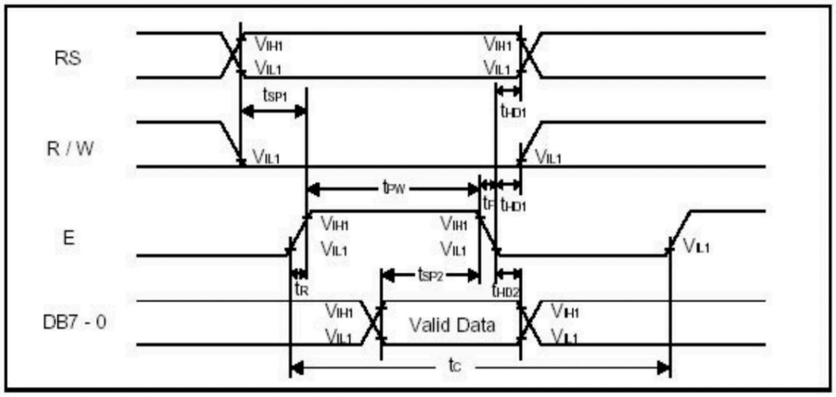


7. POWER SUPPLY

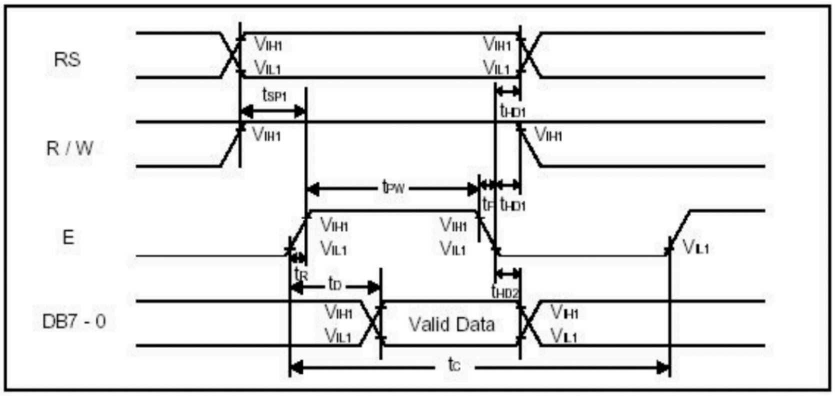


8. TIMING DIAGRAM

- WRITE OPERATION



- READ OPERATION



9. AC CHARACTERISTICS

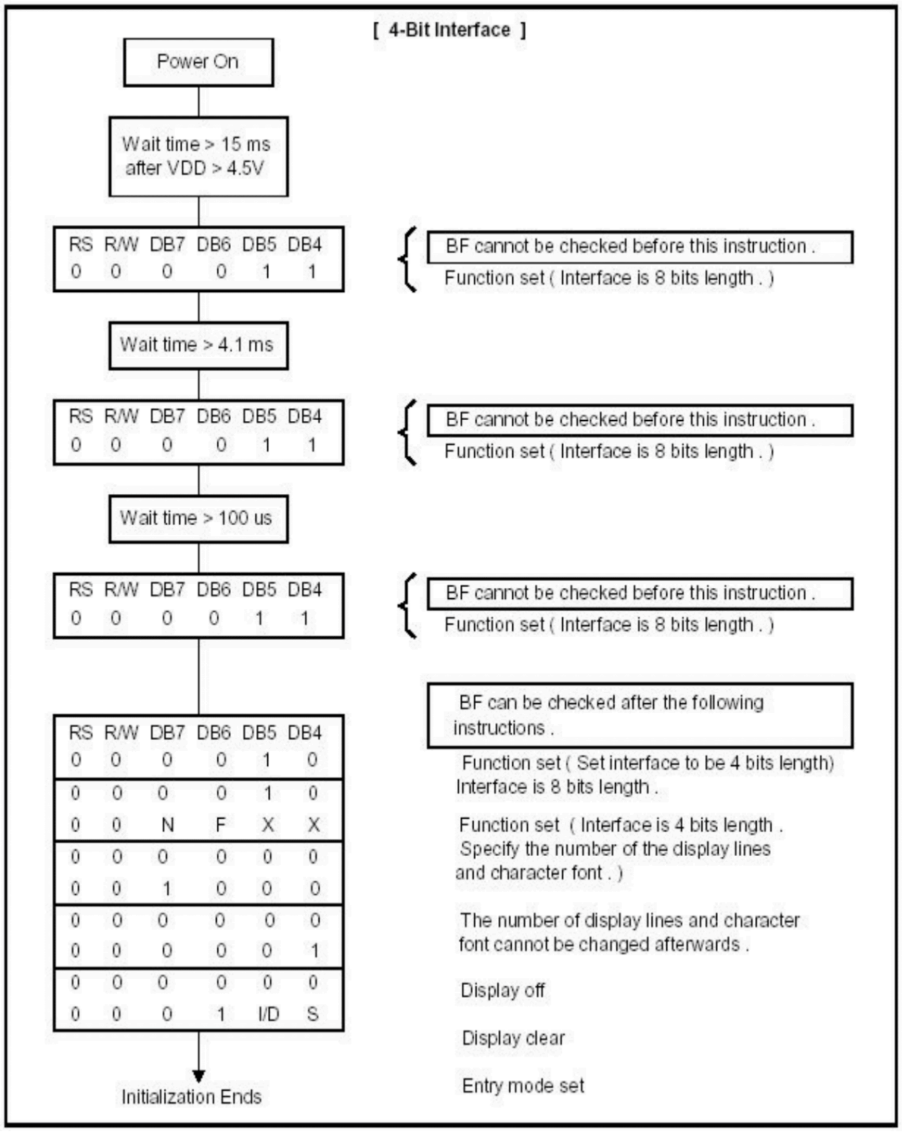
• WRITE MODE

Characteristics	Symbol	Limit			Unit	Test Condition
		Min.	Typ.	Max.		
E Cycle Time	t_C	1000	-	-	ns	Pin E
E Pulse Width	t_{PW}	450	-	-	ns	Pin E
E Rise/Fall Time	t_R, t_F	-	-	25	ns	Pin E
Address Setup Time	t_{SP1}	60	-	-	ns	Pins: RS, R/W, E
Address Hold Time	t_{HD1}	20	-	-	ns	Pins: RS, R/W, E
Data Setup Time	t_{SP2}	195	-	-	ns	Pins: DB7 - 0
Data Hold Time	t_{HD2}	10	-	-	ns	Pins: DB7 - 0

• READ MODE

Characteristics	Symbol	Limit			Unit	Test Condition
		Min.	Typ.	Max.		
E Cycle Time	t_C	1000	-	-	ns	Pin E
E Pulse Width	tw	450	-	-	ns	Pin E
E Rise/Fall Time	t_R, t_F	-	-	25	ns	Pin E
Address Setup Time	t_{SP1}	60	-	-	ns	Pins: RS, R/W,E
Address Hold Time	t_{HD1}	20	-	-	ns	Pins: RS, R/W,E
Data Output Delay Time	t_D	-	-	360	ns	Pins: DB7 - 0
Data hold time	t_{HD2}	5.0	-	-	ns	Pin DB7 - 0

10. INITIALIZATION SEQUENCE



BF cannot be checked before this instruction.
Function set (Interface is 8 bits length.)

BF cannot be checked before this instruction.
Function set (Interface is 8 bits length.)

BF cannot be checked before this instruction.
Function set (Interface is 8 bits length.)

BF can be checked after the following instructions.

Function set (Set interface to be 4 bits length) Interface is 8 bits length.

Function set (Interface is 4 bits length. Specify the number of the display lines and character font.)

The number of display lines and character font cannot be changed afterwards.

Display off

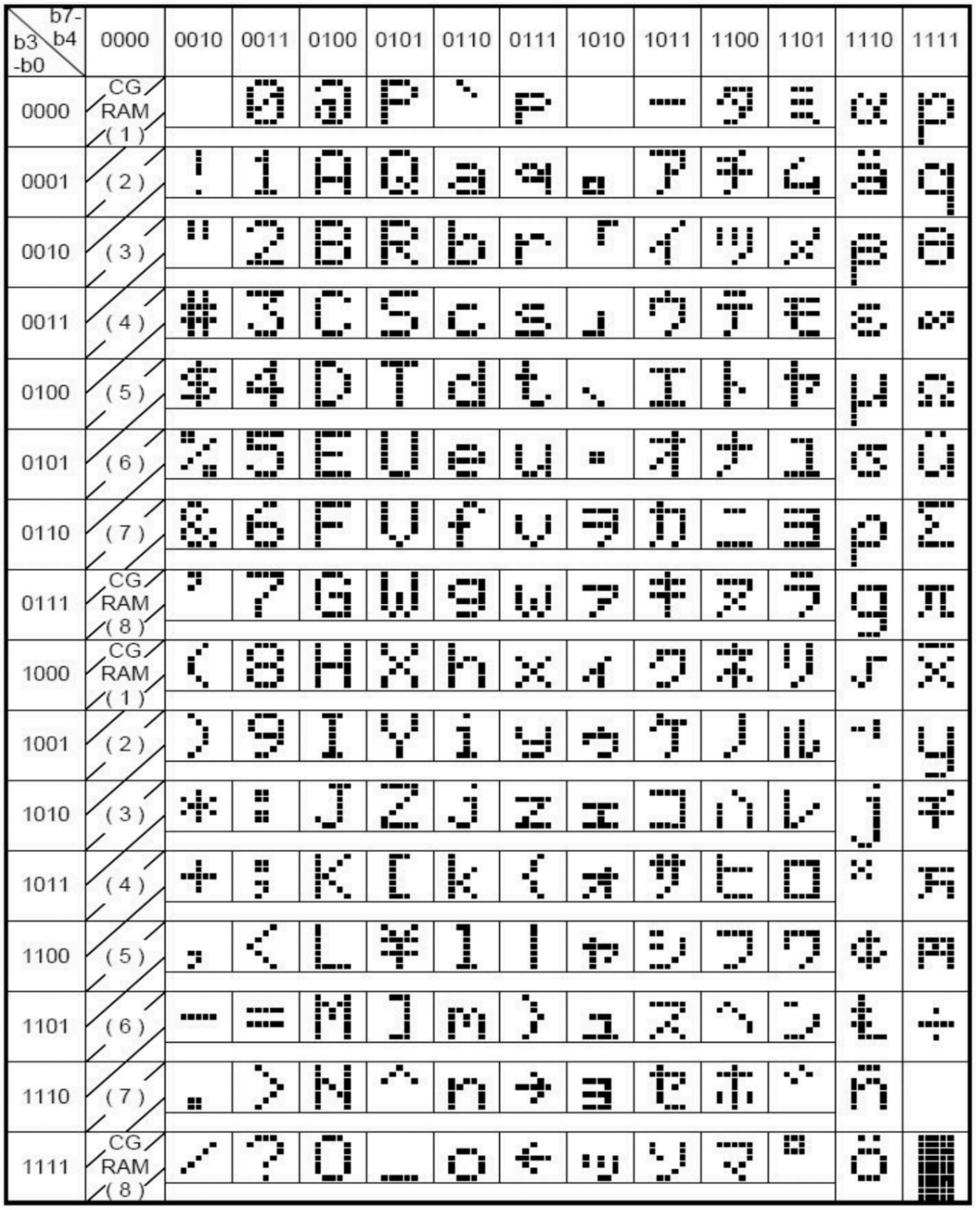
Display clear

Entry mode set

11. INSTRUCTION SET

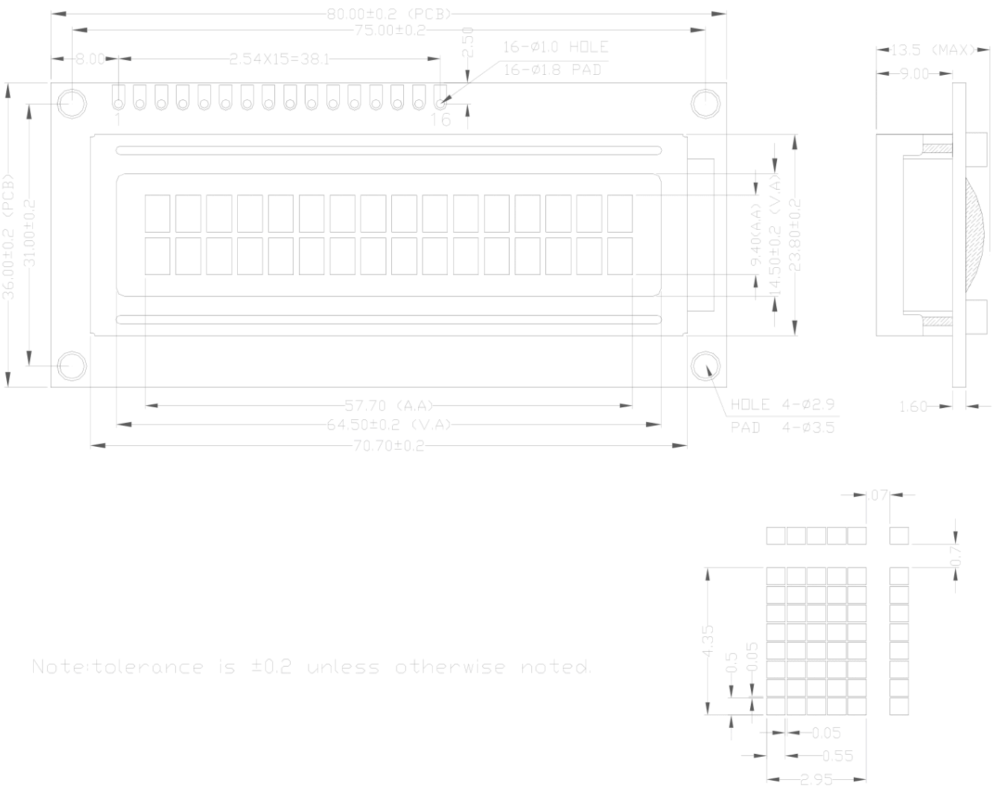
COMMAND	COMMAND CODE										COMMAND CODE	E-CYCLE f_{osc}=250KHz
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
SCREEN CLEAR	0	0	0	0	0	0	0	0	0	1	Screen Clear, Set AC to 0 Cursor Reposition	1.64ms
CURSOR RETURN	0	0	0	0	0	0	0	0	1	*	DDRAM AD=0, Return, Content Changeless	1.64ms
INPUT SET	0	0	0	0	0	0	0	1	I/D	S	Set moving direction of cursor, Appoint if move	40us
DISPLAY SWITCH	0	0	0	0	0	0	1	D	C	B	Set display on/off,cursor on/off, blink on/off	40us
SHIFT	0	0	0	0	0	1	S/C	R/L	*	*	Remove cursor and whole display,DDRAM changeless	40us
FUNCTION SET	0	0	0	0	1	DL	N	F	*	*	Set DL,display line,font	40us
CGRAM AD SET	0	0	0	1	ACG						Set CGRAM AD, send receive data	40us
DDRAM AD SET	0	0	1	ADD							Set DDRAM AD, send receive data	40us
BUSY/AD READ CT	0	1	BF	AC							Executing internal function, reading AD of CT	40us
CGRAM/ DDRAM DATA WRITE	1	0	DATA WRITE								Write data from CGRAM or DDRAM	40us
CGRAM/ DDRAM DATA READ	1	1	DATA READ								Read data from CGRAM or DDRAM	40us
	I/D=1: Increment Mode; I/D=0: Decrement Mode S=1: Shift S/C=1: Display Shift; S/C=0: Cursor Shift R/L=1: Right Shift; R/L=0: Left Shift DL=1: 8D DL=0: 4D N=1: 2R N=0: 1R F=1: 5x10 Style; F=0: 5x7 Style BF=1: Execute Internal Function; BF=0: Command Received										DDRAM: Display data RAM CGRAM: Character Generator RAM ACG: CGRAM AD ADD: DDRAM AD & Cursor AD AC: Address counter for DDRAM & CGRAM	E-cycle changing with main frequency. Example: If fcp or f_{osc}=270KHz 40us x 250/270 =37us

12. FONT TABLE



b7-b4 / b3-b0	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
0000	CG RAM (1)		0	@	P	`	p		ー	タ	ミ	α	p
0001	(2)	!	1	A	Q	a	q	。	ア	チ	ム	ä	q
0010	(3)	"	2	B	R	b	r	「	イ	ツ	メ	β	θ
0011	(4)	#	3	C	S	c	s	」	ウ	テ	モ	ε	∞
0100	(5)	$	4	D	T	d	t	、	エ	ト	ヤ	μ	Ω
0101	(6)	%	5	E	U	e	u	・	オ	ナ	ユ	σ	ü
0110	(7)	&	6	F	V	f	v	ヲ	カ	ニ	ヨ	ρ	Σ
0111	CG RAM (8)	'	7	G	W	g	w	ァ	キ	ヌ	ラ	g	π
1000	CG RAM (1)	(	8	H	X	h	x	ィ	ク	ネ	リ	√	$\bar{x}$
1001	(2)	)	9	I	Y	i	y	ゥ	ケ	ノ	ル	$^{-1}$	y
1010	(3)	*	:	J	Z	j	z	ェ	コ	ハ	レ	j	千
1011	(4)	+	;	K	[	k	{	ォ	サ	ヒ	ロ	^{x}	万
1100	(5)	,	<	L	¥	l	|	ャ	シ	フ	ワ	¢	円
1101	(6)	-	=	M	]	m	}	ュ	ス	ヘ	ン	£	÷
1110	(7)	.	>	N	^	n	→	ョ	セ	ホ	゛	ñ	
1111	CG RAM (8)	/	?	O	_	o	←	ッ	ソ	マ	゜	ö	█

13. OUTLINE DRAWING



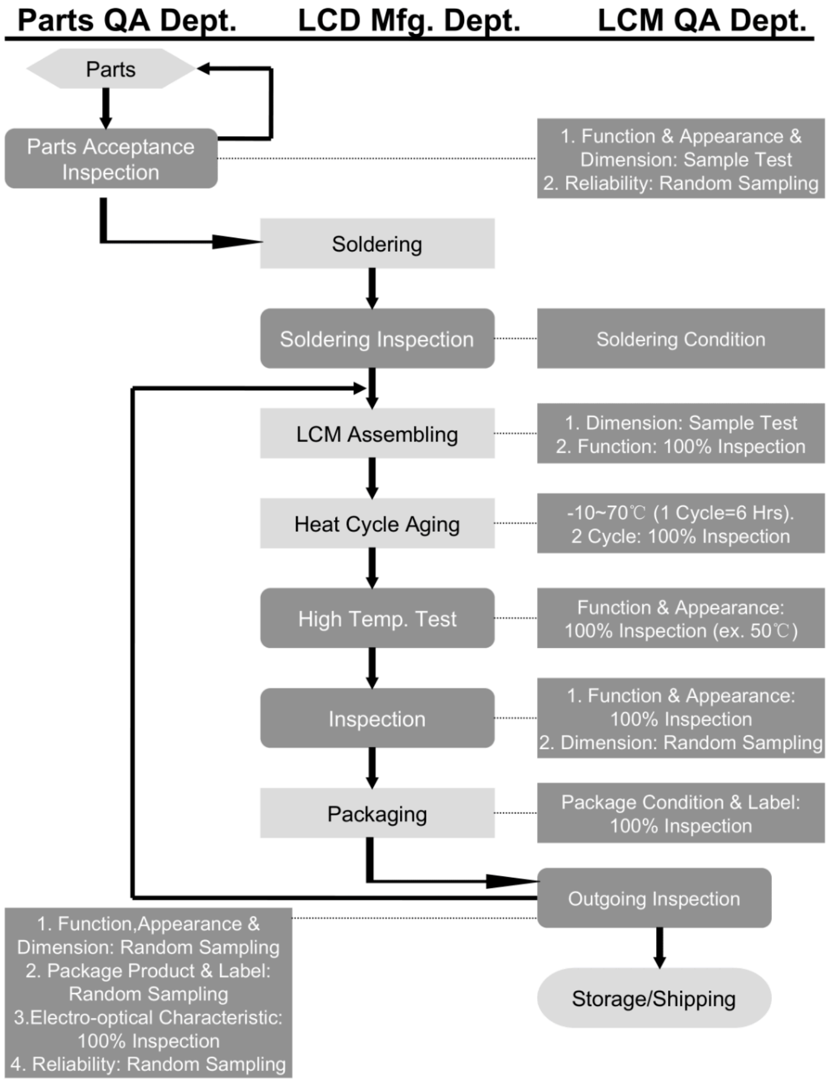
Note:tolerance is ±0.2 unless otherwise noted.

1	2	3	4	5	6	7	8
VSS	VDD	V0	RS	R/W	E	DB0	DB1
9	10	11	12	13	14	15	16
DB2	DB3	DB4	DB5	DB6	DB7	LEDA	LEDK

14. INTERFACE

PIN NO.	SYMBOL	DESCRIPTION	FUNCTION
1	VSS	GROUND	0V (GND)
2	VDD	POWER SUPPLY FOR LOGIC CIRCUIT	+5V
3	V0	LCD CONTRAST ADJUSTMENT	
4	RS	INSTRUCTION/DATA REGISTER SELECTION	RS = 0 : INSTRUCTION REGISTER RS = 1 : DATA REGISTER
5	R/W	READ/WRITE SELECTION	R/W = 0 : REGISTER WRITE R/W = 1 : REGISTER READ
6	E	ENABLE SIGNAL	
7	DB0	DATA INPUT/OUTPUT LINES	8 BIT: DB0-DB7
8	DB1		
9	DB2		
10	DB3		
11	DB4		
12	DB5		
13	DB6		
14	DB7		
15	LEDA	SUPPLY VOLTAGE FOR LED+	+5V
16	LEDK	SUPPLY VOLTAGE FOR LED-	0V

15. QC/QA PROCEDURE



16. RELIABILITY

•Operating life time:

Longer than 50000 hours (at room temperature without direct irradiation of sunlight)

•Reliability Characteristics:

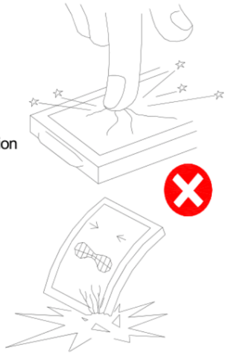
Item	Test	Criterion
High temp	50℃ / 200 Hrs	■Total current consumption should be below double of initial value ■Contrast ratio should be within initial value±50% ■No defect in cosmetic and operational function is allowable
Low temp.	0℃ / 200 Hrs	
High humidity	40℃ ＊ 90%RH / 200 Hrs	
Thermal shock	0℃→25℃→50℃→25℃ /5 Cycles (30min) (5min) (30min) (5min)	
Vibration	1.Operating time: Thirty minutes exposure in each direction (x, y, z) 2.Sweep Frequency (1min):10Hz→ 55Hz →10Hz 3.Amplitude: 0.75mm double amplitude	

17. Handling Precautions

1. Limitation of Application:
Jing Handa products are designed for use in ordinary electronic devices such as business machines, telecommunications equipment,measurement devices and etc. Please handle the products with care. (see below)
Jing Handa products are not designed,intended ,or authorized for use in any application which the failure of the product coul result in a situation where personal injury or death may occur . these applications include, but are not limited to . life-sustaini equipment,nuclear control devices , aerospace equipment , devices related to hazardous or flammable materials , etc.[If Buy intends to purchase or use the Jing Handa Products for such unintended or unauthorized applications , Buyer must secure p written consent to such use by a responsible officer of Jing Handa Corporation.]Should Buyer purchase or use Jing Handa I any such unintended or unauthorized application [without such consent].Buyer shall indemnify and hold Jing Handa and its employees. subsidiaries, affiliates and distributors harmless against all claims, costs, damages and expenses , and reasonab attorney's fees,arising out of , directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Jing Handa was negligent regarding the design or manufacture of the part.
2.Industrial Rights and Patents
Jing Handa shall not be responsible for any infringement of industrial property rights of third parties in any country arising ou application or use of Jing Handa products, except which directly concern the structure or production of such products..

No Press and Shock!

If pressure to LCD, orientation may be disturbed.
LCD will broken by shock!

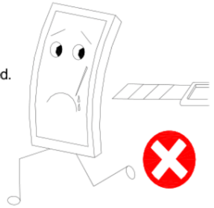


Don't Swallow or Touch Liquid Crystal!

Liquid Crystal may be leaked when display is broked.
If it accidentally gets your hands, wash then with water!

Don't not Scratch!

Polarizer is a soft material and can easily be scratched.

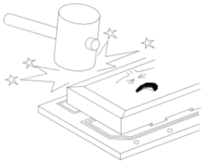


No DC Voltage to LCD!

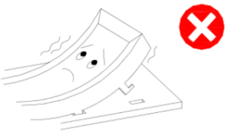
DC volrage or driveing higher than the specified voltage will reduce the lifetime of the LCD.

Don't Press the Metallic Frame and Disassem the LCM

Pressure on the metallic frame and PCB may deform the conductive rubber or break the liquid crystal cell and back light, which will cause defects.



LCD may be shifted or conductive rubber may be reshaped, which will cause defects.

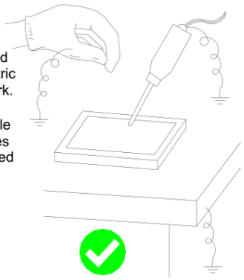


Slowly Peel Off Protective Film!

Avoid static electricity.

Avoid Static Electricity!

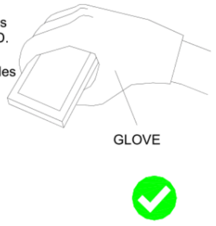
Please be sure to ground human body and electric apploances during work.
It is preferable to use conductive mat on table and wear cotton clothes or conduction processed fiber. Synthetic fiber is not recommended.



Wear Gloves While Handing!

It is preferable to wear gloves to avoid damaging the LCD.

Please do not touch electrodes with bare hands or make them dirty.



Keep Away From Extreme Heat and Humidity

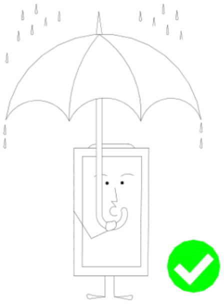
LCD deteriorates.

Use Alcohol to Clean Terminals!

When attaching with the heat seal or anisontropically conductive film, wipe off with alcohol before use.

Don't Drop Water on LCD!

Note that the presence of waterdrops or dew in the LCD panel may deteriorate the polarizer or corrade electrode.



Precaution in Soldering LCD Module

Basic instructions: Solder I/O terminals only.
Use soldering iron without leakage.
(1)Soldering condition to I/O terminals
Temperature at tip of the iron: 280±10℃
Soldering time: 3~4 sec.
Type of solder: Eutectic solder (containing colophony-flux)
*Please do not use flux because it may soak into LCD Module or contaminate it.
*It is preferable to peel off protective film on display surface after soldering I/O terminals is finished.
(2)Remove connector or cable
*When you remove connector or cable soldered to I/O terminals, please confirm that solder is fully melted. If you remove by force, electrodes at I/O terminals may be damaged(or stripped off).
*It is recommended to use solder suction machine.

Long-term Storage

If it is necessary to store LCD modules for a long time, please comply with the following procedures.
If storage condition is not satisfactory, display(especially polarizer) may be deteriorated or soldering I/O terminals may become difficult(some oxide is generated at I/O terminals plating).

1.Store as delivered by Jing Handa
2.If you store as unpacked,put in anti-static bag,seal its opening and store where it is not subjected to direct sunshine nor fluorescent lamp.
3.Store at temperature 0 to +35℃ and at low humidity.Please refer to our specification sheets for storage temperature range and humidity condition.

Long-term Storage

Please use power supply with built-in surge protection circuit.

Works Cited

Amin, S., Achenbach, S. J., Atkinson, E. J., Khosla, S., & Melton, L. J., 3rd (2014). Trends in fracture incidence: a population-based study over 20 years. Journal of bone and mineral research : the official journal of the American Society for Bone and Mineral Research, 29(3), 581–589. https://doi.org/10.1002/jbmr.2072

Arduino Team. (2021, December 9). One board to rule them all: History of the Arduino UNO. Arduino Blog. https://blog.arduino.cc/2021/12/09/one-board-to-rule-them-all-history-of-the-arduino-uno/

Casavola, C., Pappalettera, G., Pappalettere, C., Patronelli, M., Renna, G., Laurenziello, M., & Ciavarella, D. (2022). A full-field DIC analysis of the mechanical-deformation behavior of polyethylene terephthalate glycol (PET-G) aligners. *Journal of the Mechanical Behavior of Biomedical Materials*, *134*, 105391. https://doi.org/10.1016/j.jmbbm.2022.105391

Cast Care 101: 10 Tips for Surviving Discomfort. (2019, April 16). *Chester County Hospital*. April 24, 2024, https://www.chestercountyhospital.org/news/health-eliving-blog/2019/august/cast-blog-article

Gordon, A. M., Malik, A. T., & Goyal, K. S. (2021). Trends of hand injuries presenting to US emergency departments: A 10-year national analysis. *The American Journal of Emergency Medicine*, *50*, 466–471. https://doi.org/10.1016/j.ajem.2021.08.059

Hafen BB, Sharma S. Oxygen Saturation. [Updated 2022 Nov 23]. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing; 2023 Jan-. Available from: https://www.ncbi.nlm.nih.gov/books/NBK525974/

Huggenberger R., & Detmar M. (2011, Dec.). The Cutaneous Vascular System in Chronic Skin Inflammation, J Investig Dermatol Symp Proc, vol. 15, no. 1, pp. 24–32.

IEEE (2022). IEEE Standard for Wearable Consumer Electronic Devices--Overview and Architecture, IEEE Std 360-2022, vol., no., pp.1-35, 25, 10.1109/IEEESTD.2022.9762855.

Jena, Sudarsana & Gupta, Ankur. (2021). Review on pressure sensors: a perspective from mechanical to micro-electro-mechanical systems. Sensor Review. 10.1108/SR-03-2021-0106].

Jubran A. (1999). Pulse oximetry. *Critical care (London, England)*, *3*(2), R11–R17. https://doi.org/10.1186/cc341

Kowalski, Kurtis & Pitcher, J & Bickley, Barry. (2002). Evaluation of Fiberglass versus Plaster of Paris for Immobilization of Fractures of the Arm and Leg. Military medicine. 167. 607-61. 10.1093/milmed/167.8.607.

Mayo Clinic (2019) Hand and Wrist Fractures *The Mayo Clinic* https://sportsmedicine.mayoclinic.org/condition/hand-wrist-fractures/#:~:text=Hand%20and%20wrist%20fractures%20are,the%20 bones%20of%20your%20 wrist

Measurements Group, Inc. (1988). Strain Gage Based Transducers: Their Design and Construction (Ser. 0-9619057-0-0).

NOAA Office of Response and Restoration, U. G. (2009). *Search Chemicals*. NOAA. https://cameochemicals.noaa.gov/chemical/25054#:~:text=PLASTER%20OF%20PARIS%20is%20non,water%20to%20form%20gypsum%20CaSO4

Ohio State University (2022). Arm Injury Statistics | Aids for One Armed Tasks. Arm Injury Statistics Ohio State University https://u.osu.edu/productdesigngroup3/sample-page/#:~:text=Fractures%20occur%20at%20an%20annual,women%20(2.0%20per%20100).&text=After%20age%2045%2C%20however%2C%20fracture,higher%20among%20women%20than%20men

Pirman, F., & Awang, A. (2022). Enhanced mechanical properties plaster of Paris with addition of rice husk fibers. IOP Conference Series: Earth and Environmental Science, 1103(1), 1–7. https://doi.org/10.1088/1755-1315/1103/1/012008

Porter, M. E., & Heppelmann, J. E. (2020, September 10). *How smart, Connected Products Are Transforming Competition*. Harvard Business Review. https://hbr.org/2014/11/how-smart-connected-products-are-transforming-competition

Rafi BM, Tiwari V. (2023, Aug, 8). Forearm Fractures. StatPearls [Internet]. Treasure Island (FL): StatPearls https://www.ncbi.nlm.nih.gov/books/NBK574580/

SainSmart. (2024). All colors, TPU flexible filament 1.75mm 0.8kg/1.76lb. SainSmart.com. https://www.sainsmart.com/collections/tpu-filament/products/all-colors-tpu-flexible-filament-1-75mm-0-8kg-1-76lb

Schlégl, Ádám & Told, Roland & Kardos, Kinga & Szőke, András & Ujfalusi, Zoltán & Maroti, Peter. (2022). Evaluation and Comparison of Traditional Plaster and Fiberglass Casts with 3D-Printed PLA and PLA–CaCO3 Composite Splints for Bone-Fracture Management. Polymers. 14. 3571. 10.3390/polym14173571.

Simmons, E. H., & Cox, L. A. (1957). A clinical and experimental study of plaster of Paris bandages in Canada. Canadian Medical Association journal, 76(11), 941–946.

Son, Y. S., & Kwon, K. H. (2023). Utilization of smart devices and the evolution of customized healthcare services focusing on big data: a systematic review. *mHealth*, *10*, 7. https://doi.org/10.21037/mhealth-23-24

Standard guide for evaluation of thermoplastic polyurethane solids and solutions for biomedical applications. F624. (n.d.). https://www.astm.org/f0624-09r15e01.html

Szostakowski, B., Smitham, P., & Khan, W. S. (2017). Plaster of Paris-Short History of Casting and Injured Limb Immobilization. The open orthopedics journal, 11, 291–296. https://doi.org/10.2174/1874325001711010291

Silverio, Manuel & Renukappa, Suresh & Suresh, Subashini. (2018). What is a smart device? - a conceptualisation within the paradigm of the internet of things. Visualization in Engineering. 6. 10.1186/s40327-018-0063-8.

Tower Pro (2024). *SG92R Tower Pro.* Retrieved from https://www.towerpro.com.tw/product/sg92r-7/.

Tuan, C. C., Lu, C. H., Wu, Y. C., Yeh, W. L., Chen, M. C., Lee, T. F., Chen, Y. J., & Kao, H. K. (2019). Development of a System for Real-Time Monitoring of Pressure, Temperature, and Humidity in Casts. Sensors (Basel, Switzerland), 19(10), 2417. https://doi.org/10.3390/s19102417

Wright Jr., John R. (2022). Introduction to Microcontrollers [Paper presentation], AENG 467, Mobile Robotics, Millersville, Pennsylvania. https://sites.millersville.edu/jwright/L1%20AENG%20467%20Intro%20to%20the%20Microcontroller%202022.pdf

Xu, T., Shen, W., Lin, X., & Xie, Y. M. (2020). Mechanical Properties of Additively Manufactured Thermoplastic Polyurethane (TPU) Material Affected by Various Processing Parameters. Polymers, 12(12), 3010. https://doi.org/10.3390/polym12123010/