GESTURE WATCH

A Technical Paper submitted to the Department of Electrical & Computer Engineering
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Engineering

By
Pearak “Derek” Tan

April 28, 2020

Technical Project Team Members
Julian Nguyen
Edward Ryan

ADVISOR
Harry Powell, Department of Electrical & Computer Engineering

Watchmen / Gesture Watch

Pearak "Derek" Tan, Julian Nguyen, Edward Ryan

12/16/2019

Capstone Design ECE 4440 / ECE4991

Signatures

[illegible]

[illegible]

Eddie Ryan IV

Statement of work

Julian was the primary hardware engineer, designing the circuit schematics in Multisim as well as the PCB layout and custom part footprints in Ultiboard. He used his electrical engineering expertise to choose the proper components and part values needed to procure through DigiKey/Mouser orders. Julian was also responsible for all testing/debugging involving electronics and hardware, and was additionally the secondary person helping with 3D printing and assembly.

Derek was the primary designer developing the chassis/housing for the watch. He used his previous experience in 3D modeling and printing to ensure that all components of the watch were properly and professionally enclosed. He also served as the secondary programmer in the project and worked on the LED driver code, and served as the secondary hardware debugger/tester and verified all functionalities specified in the test plan.

Edward was the primary software engineer/programmer of the project. He used his prior experience in embedded C and Code Composer to develop firmware to act as an effective gateway between the hardware and software components. Edward wrote software for both the watch's behavior and the Bluetooth interface. He also wrote the AutoIt driver for the receiving Bluetooth device.

Table of Contents

Table of Figures

Abstract

Our project consists of making a simple wearable device that is completely enclosed and can act as a remote motion controller. The watch contains an MSP430F1611 microprocessor [1] and is able to connect to a computing device via Bluetooth [2]. The main functionality of the watch is gesture-based control, a feedback system that notifies the user when actions are registered, and a rechargeable battery that allows the watch to be freely used for at least several hours. To achieve this, we will require 4 main component subsystems, the MCU/communications module (i.e. MSP430 and Bluetooth adapter), the feedback system (i.e. vibration motor and LED display), an actuator/sensor input (pushbutton and gyroscope/accelerometer), and a power module (i.e. battery and charging module). This watch was also designed in a form factor small enough to fit comfortably on a user's wrist during use.

Background

We chose this project because we wanted to create a project that was challenging as well as usable as an everyday wearable device with a unique feature. Many companies have produced their own smartwatches with their own features, and some hobby kits and personal smartwatch projects exist such as one created by S. March [3]. This project created by March had very limited functionality, with no gesture controls. This project also utilizes a DA14683 [4] microcontroller, which for our purposes will not be feasible due to the form factor being too small. Advanced Circuits [5] is unable to create a PCB that accommodates a microcontroller this small while still meeting the limitations of the student's special boards. The main differentiating feature of our smartwatch is the ability to interpret complex gesture controls in an intuitive and responsive way. For example, making a sweeping motion in some direction while wearing the watch causes a slide presentation to move forwards or backwards. Other projects incorporate basic accelerometer logging, button input, and a clock. We took inspiration from past smartwatch projects such as the one done by S. March [3] and will use it as a resource going forward. This project called upon our knowledge from all of the ECE Fundamentals classes [6], mainly in the form of circuits and PCB design. It required all our knowledge from our previous coding classes, mainly CS 2150 [7] and ECE 3430 [8] for their teachings in C/C++ [9] and lower/embedded systems.

Constraints

Design Constraints

The foremost design constraints for our project were the size and power limitations that we had to design around in order to allow the final device to fit in a comfortably wearable form factor. Thus, we committed to a PCB restriction of 40mm in diameter to have typical watch face dimensions. In order to fit our desired schematic onto such a small PCB surface, we chose components with appropriately small footprints and low power consumption. The majority of chips and components fixed to the PCB were surface-mount devices (SMD) [10] to keep high circuit density and as low a profile as possible. This allowed us to keep our PCB very compact for the final product, similar to professional microelectronics.

Although many wearable and mobile devices consist of extremely space-efficient surface-mount components and integrated chips, we were restricted to using only moderately small parts: namely larger 0603 package resistors/capacitors and SOT [11] package transistors. This was due to Advanced Circuit's manufacturing limits of at least 5-mil trace width for PCB connections. These size requirements were also reinforced by the limitations of 3W's part mounting service.

Our choice of lithium-ion battery was also limited by manufacturing and shipping capabilities, as procurement of batteries was restricted to more delayed ground shipping for safety and cost. We settled for a smaller single 85mAh coin battery that was predicted to be sufficient to power the device for approximately 2-3 hours of regular usage.

Economic and Cost Constraints
As our project was inspired by industry/consumer smartwatches, we strived to keep the cost of components and manufacturing within $200. This was possible due to our aforementioned design which used cheaper and popular SMD electronic components through bulk DigiKey [12] and Mouser [13] orders. Our chosen scale of larger SMD packages also prevented the need for more specialized equipment or manufacturing processes when it came to mounting and assembly.

However, since our final design still relied on small surface-mount parts and chips, our testing/verification for various subsystems had to be done through iterations of fully mounted prototype PCBs. We often depended on 3W's services to assemble each prototype PCB needed for a round of schematic updates. Due to the course's limited PCB order schedule and restrictions on duplicate part orders, we had to carefully plan and ration batches of PCBs and parts in order to comprehensively mount and test at each design stage.

External Standards
The appropriate safety methods were taken into account when working with circuits and soldering the PCB. Appropriate safety wear was used in both testing and assembly. The watch used Bluetooth 2.1 [2] to connect to a computer. It is commonly used for wireless headphones and other audio devices, as well as wireless keyboards, mice, and game controllers. Bluetooth is also used for communication between various smart home and Internet of Things (IoT) devices. The lithium-ion battery that is used to provide power to the watch also meets the standard used to ensure power retention and high energy density. Standard PLA [14] plastic is used to 3D print the chassis due to its affordability and finish quality, the Windows 10 operating system [16] was used to run all the software and was the primary development environment for our MSP430 [1] chip. The MISRA C [17] (Motor Industry Software Reliability Association) coding standard will be used to ensure consistency and readability.

Tools Employed
Software tools used to design and build the watch include:

1. Code Composer [18] - IDE used to develop, debug, and upload programs to MSP430
2. C programming language [9] - Code Composer language
3. Multisim [19] - used for circuit schematic design, simulation, and parts inventory
4. Ultiboard [20] - used to design custom part footprints and PCB layout
5. Inventor [21] - CAD used to 3D model the watch chassis and charger housing
6. Cura [22] - used to slice 3D models into a format for 3D printing
7. RealTerm [23]- networking terminal used to debug Bluetooth connectivity
8. Autoit [24] - used as Windows driver for interpreting Bluetooth commands

Ethical, Social, and Economic Concerns

Environmental Impact

The primary environmental concern of our project is the power module of the device, as lithium-ion batteries used by mobile and wearable devices have a significant energy/pollutant footprint. We needed to ensure that our design efficiently managed power, specifically that it was not wastefully drawn when charging and the watch was able to fully function for a sufficient amount of time before draining the battery.

Sustainability

Our sustainability concerns also stem from the efficiency and integrity of the battery/power system and thus, that of the entire device. Proper battery protection needed to be implemented in order to prevent the battery from both overcharging and overdraining. Both cases of charging the battery cell past its intended capacity and draining the battery to dangerously low voltages would lead to long term damage and degradation of the primary power supply to the system, resulting in decreasing functionality and lifetime of the device.

Health and Safety

The power module of the watch also poses various safety concerns. The charging contact and battery are potential threats of exposure of electrical faults to the wearer of the watch. We had to make sure that the device was fully enclosed so that human skin would never come in contact with exposed circuitry. In addition to ensuring that the watch's power management was efficient, we also had to ensure that there was no major risk of injuring the user. A battery protection circuit was necessary to keep the battery from overcharging and thus overheating and harming the wearer. The feedback system components may also pose a minor health risk if vibration or LED brightness intensity are not properly regulated to accommodate sensitive users. We also had to consider whether the plastic filament used to build the watch housing posed no material/chemical health risk.

Manufacturability

Due to the smaller wearable form factor, assembly would be an overarching challenge. However, for the same reason, the demand for excessive or expensive components was limited to a manageable scale. The design would mainly consist of common electronic components that were readily available from DigiKey [8] or other popular hardware suppliers. Non-electronic components were equally accessible, with the watch housing being 3D-printable and watch bands being a standardized and interchangeable component. The restricted scale of components, PLA filament [9], and other parts estimated a total cost no more than $200. All in all, the watch is highly manufacturable since all of the components are on popular hardware suppliers and no custom parts were used except for the chassis, which could be easily molded to produce in bulk in a manufacturing setting.

Ethical Issues
There could be privacy and transparency concerns in regards to the fact that the watch is considered “smart” enough to potentially aid with cheating on exams or other regulated activities. Since the watch has Bluetooth capabilities and intuitive gesture controls, it is possible for a user to utilize the watch’s visual and tactile feedback system to participate in discreet and unauthorized communication.

Intellectual Property Issues
We believe that some aspects of the watch are patentable, while others are not. The things that are not are the gestures that are already patented by other companies. We can see that in US Figure 2: Patent 20160299570 [25] by Apple, a gesture that we were using for start/stop has already been patented. This gesture is the one that involves the rotation of the wrist. While this gesture has been already patented, we believe that our other gestures have merit and are patentable. There are also similar products that achieve the same purpose as our gesture watch through slightly different means. Our driver interface that correlates a gesture with a keyboard command can also be patented. Figure 1: US Patent US9582076B2 [26] by Microsoft features a ring that connects to a smart device and performs actions on that device depending on the gesture detected.

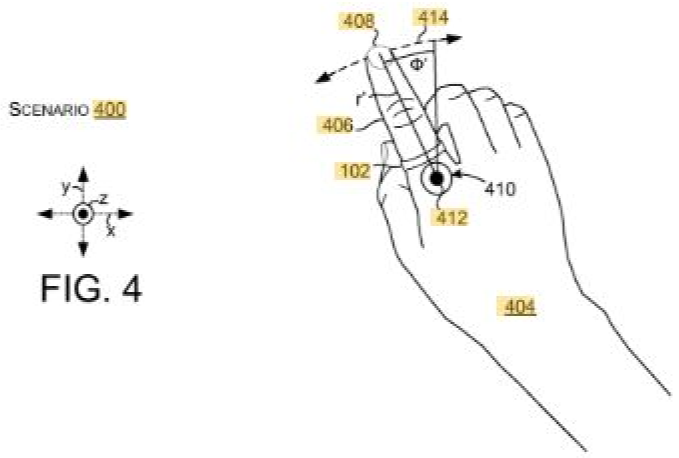


FIG. 4

Figure 1: US Patent US9582076B2 by Microsoft

We do not infringe upon any of the claims made by Microsoft in US Patent US9582076B2 [27]. The smart gesture ring above claims that it is a system, comprising:

- a Smart ring configured to be worn on a first segment of a finger of a user;
- at least one flexion sensor secured to the Smart ring in a manner that can detect a distance between the at least one flexion sensor and a second segment of the finger;
- and, an input component configured to: analyze signals from the at least one flexion sensor to detect an initialization sequence, in response to the detected initialization sequence, calibrate a coordinate system of the Smart ring, and use the calibrated coordinate system to interpret further signals from the at least one flexion sensor.

Although Microsoft's device performs a similar function, none of our features infringe on their independent claims.

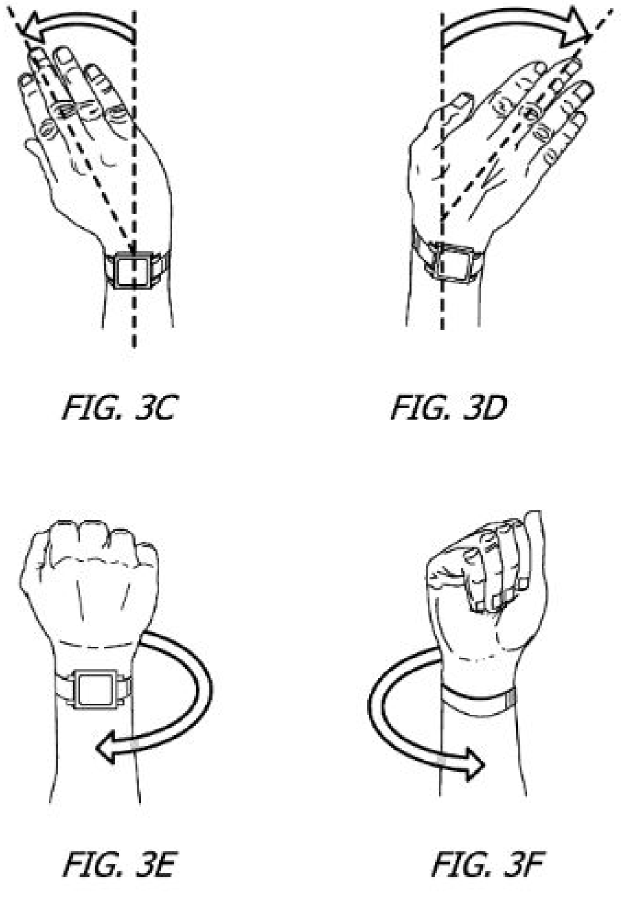


Figure 2: US Patent 20160299570 by Apple

An independent claim on Figure 2: US Patent 20160299570 [25] by Apple involves a method of operating a wrist-worn device, the method comprising:

- detecting, using a sensor on the wrist-worn device, a force acting on a band element of the wrist-worn device, the force indicative of a wrist articulation;
- determining, using data from an accelerometer in the wrist-worn device, whether the user's arm is in motion;
- interpreting, by a processing Subsystem of the wrist-worn device, the detected force as corresponding to a wrist gesture, wherein the interpretation is based in part on the detected force and in part on whether the user's arm is in motion
- and invoking a function of the wrist-worn device based on the wrist gesture

Our watch does all the things in the independent claim above, but it does not just use the accelerometer data to perform a gesture, it primarily uses a gyroscope. The patent also makes the dependent claim that "the method of claim 1 wherein the wrist gesture includes a dorsiflexion of the wrist." Our stop and start gesture may infringe this claim.

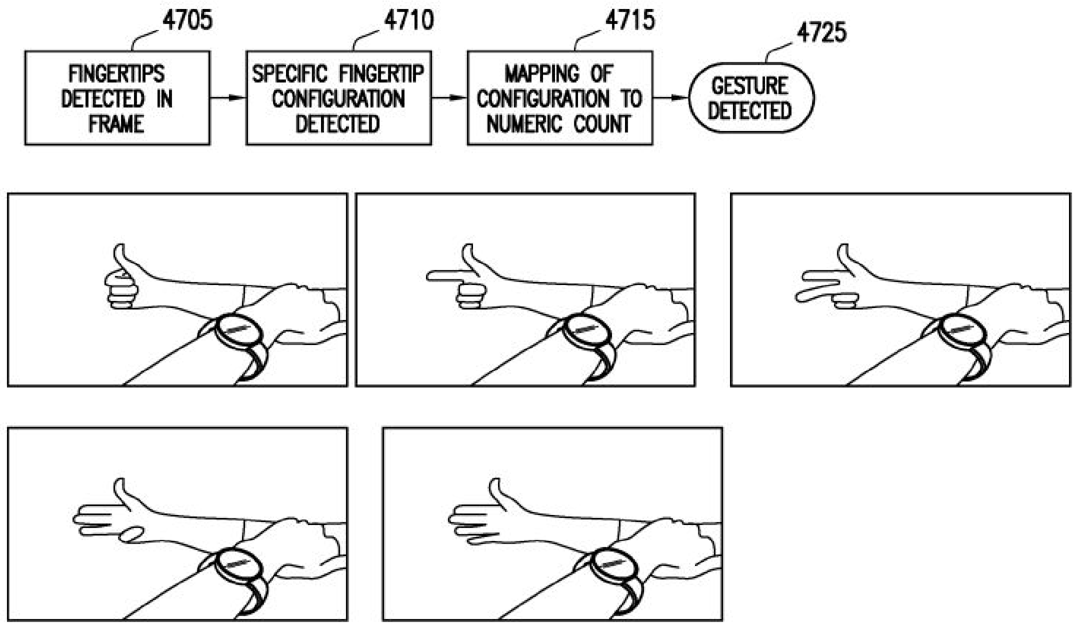

4705
4710
4715
4725
FINGERTIPS DETECTED IN FRAME
SPECIFIC FINGERTIP CONFIGURATION DETECTED
MAPPING OF CONFIGURATION TO NUMERIC COUNT
GESTURE DETECTED

FIG. 47

Figure 3: US Patent Application 14/014,940 by Samsung

We do not infringe upon any of the claims made by Microsoft in US Patent Application 14/014,940. The above patent makes an independent claim that the watch is "an apparatus comprising one or more processors and a memory," while it is also able to execute a series of instructions. It then expands on the claim and explains that the watch analyzes the task of an application, delegates the task to be processed by one or more computing devices, and receives the computing devices results from processing the delegated task. None of the features in our Gesture Watch infringe on these claims. While our watch does have a processor and a memory, it does not analyze the task of an application, it only serves to send windows keyboard commands to a connected device.

Detailed Technical Description of Project

Our project is a wearable smartwatch device with the aforementioned functionality of having thorough motion/orientation tracking, allowing the ability to implement software-defined gesture controls by interpreting motion changes. In addition, the device will have the timekeeping functionality of typical watches displayed through an LED face. Our project works by integrating the various subsystems detailed below (Figure 1.1) and soldering them onto a specialized small PCB that we design. The PCB will be powered by a lithium-ion battery which is rechargeable by a wireless inductive charging receiver. All of these hardware components will be enclosed in a 3D-printed watch body housing with standard watch band pins [26]. The watch's software and firmware will be developed to allow a user to pair the watch with their personal computing device via Bluetooth [2], such that it may act as a remote controller for the paired device. The figure below is a general block diagram for all of the subsystems and constituent parts of the watch, including the specific devices we used.

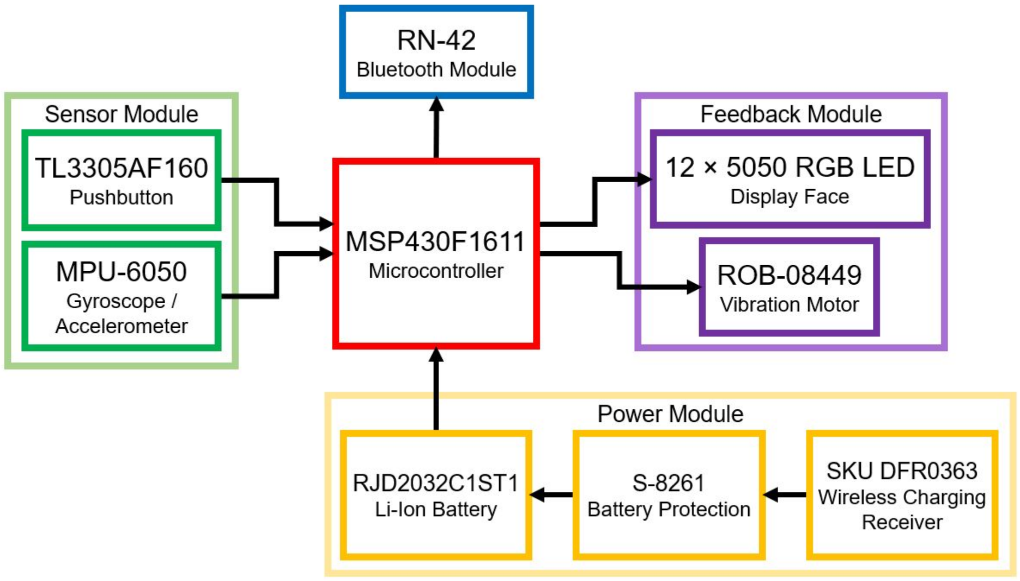


Figure 4: Hardware Subsystem Overview Block Diagram

The project’s main subsystems are laid out in Figure 1.1 above. These systems consist of the sensors for user input (green), central processor and communications (red/blue), device power (yellow), and feedback devices (purple).
The sensor module firstly has a simple pushbutton switch [27] that allows the user to toggle between different behavior modes (detailed in Figure 1.2). It also includes an MPU-6050 inertial measurement unit (IMU) [28] that senses gyroscopic and acceleration data to be processed for motion/gesture detection.
The MSP430 microprocessor [1] acts as the central controller for all sensor, communication, and feedback devices, with the RN-42 module [29] being both a communicator and antenna for interfacing the MSP430 with other devices via Bluetooth.
The RJD lithium-ion battery [30] powers all devices and is rechargeable via the SKU DFR0363 [31] wireless inductive charging receiver. The S-8261 battery protection integrated circuit [32] acts as an intermediary between the battery and charging receiver to prevent undesired discharge and harmful under/overvolting.
An array of 5050 RGB LEDs [33] form a 12-segment display face for indicating time as well as visualizing motion control. The ROB-08449 [34] is a small vibration motor that is activated for haptic feedback to the wearer whenever a gesture is registered.

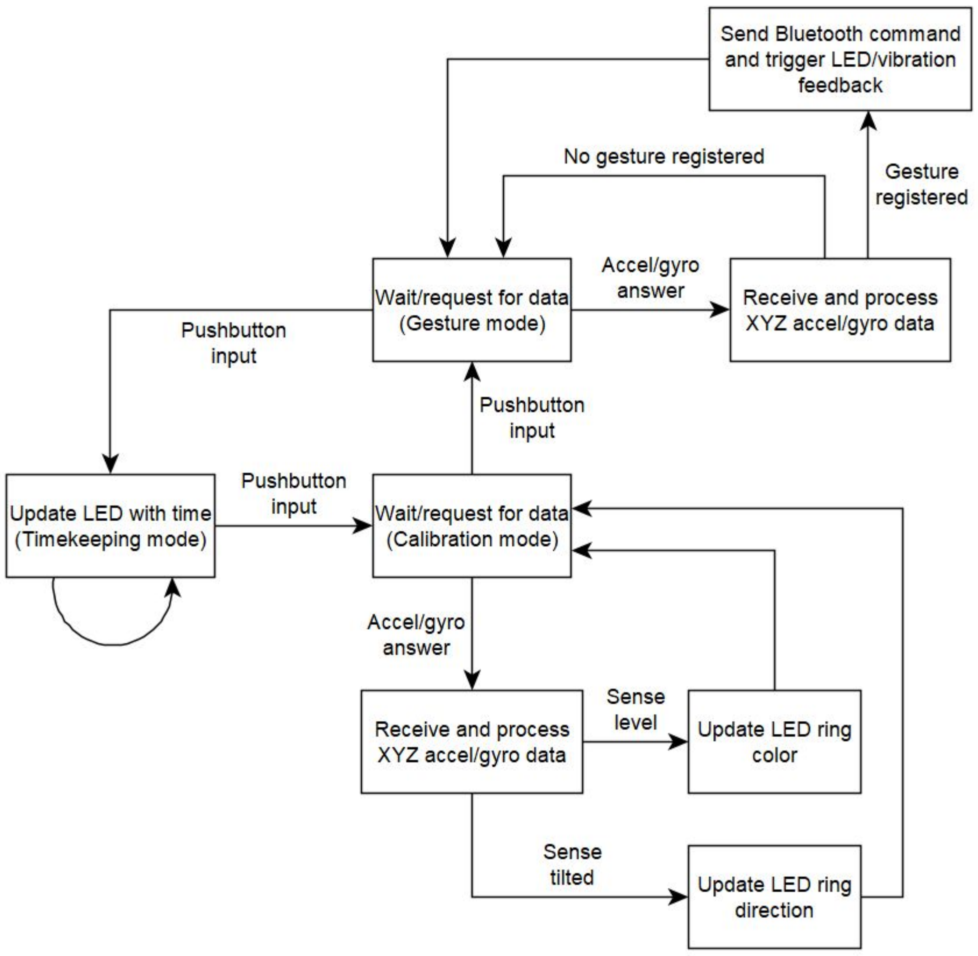

Send Bluetooth command and trigger LED/vibration feedback
No gesture registered
Gesture registered
Accel/gyro answer
Wait/request for data (Gesture mode)
Receive and process XYZ accel/gyro data
Pushbutton input
Pushbutton input
Update LED with time (Timekeeping mode)
Pushbutton input
Wait/request for data (Calibration mode)
Accel/gyro answer
Receive and process XYZ accel/gyro data
Sense level
Update LED ring color
Sense tilted
Update LED ring direction

Figure 5: Watch Behavior State Machine

Figure 1.2 illustrates the software behavior of the watch. Whenever the user presses the pushbutton, the watch toggles between 3 predefined behavior modes: standard timekeeping, motion calibration, and gesture control.

1. Timekeeping mode consists of the LED display face continuously updating with the current time by lighting up segments of the array corresponding to hours, minutes, and seconds.
2. Calibration mode allows the user to visualize the watch's motion detection by indicating different orientations through the LED face display. When the watch is held at level (approximately 0 degrees), the display face will light up green. When the watch is tilted on its side at some axis, the display face will show a blue light indicator in the corresponding direction.
3. In gesture control mode, motion data is interpreted from the gyroscope/accelerometer whenever a movement is detected. If the recorded motion is sufficient for the gesture algorithm to detect as a valid gesture, a corresponding Bluetooth command is transmitted and both LED and vibration feedback are activated to notify the user.

Debugging Components Used

Part Name	**Schematic**
MSP430F1611 Header Board [35]	https://www.olimex.com/Products/MSP430/Header/_resources/MSP430-Hxxx-e.pdf
SparkFun MPU-6050 Header Board [36]	https://cdn.sparkfun.com/datasheets/Sensors/IMU/Triple_Axis_Accelerometer-Gyro_Breakout_-_MPU-6050_v12.pdf
SparkFun RN-42 Header Board [37]	https://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/BlueSMiRF-Gold-ChipAnt-v1_rotat2.pdf

Figure 6: Debugging Components Used

Figure 6 indicates the components that were used for prototyping and debugging. These were all header boards that helped us in preliminary coding and hardware verification before working with the final PCB design.

Components Used

Part Name	**Part Type**	**Datasheet/Part Link**
MSP430F1611 [38]	Microcontroller	http://www.ti.com/lit/ds/symlink/msp430f155.pdf
RN-42 [39]	Bluetooth Module	http://ww1.microchip.com/downloads/en/DeviceDoc/50002328A.pdf

MPU-6050	Gyroscope/Accelerometer	http://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
ROB-08449	Vibration Motor	https://cdn.sparkfun.com/datasheets/Robotics/B1034.FL45-00-015.pdf
NeoPixel LED Ring	LED Ring	https://cdn-shop.adafruit.com/product-files/1138/SK6812+LED+datasheet+.pdf
SKU DFR0363	Wireless Charging Transceiver and Receiver (5V/300mA)	https://www.dfrobot.com/product-1283.html?gclid=Cj0KCQjwoKzsBRC5ARIsAITcwXEspswIvyiB1Q6sAQh-SAp9RjlocUqFxZQUHykKQ3fFlpWXdAISqi0aAjKLEALw_wcB
RJD2032C1ST1	Lithium-ion Battery (Rechargeable)	https://www.digikey.com/product-detail/en/illinois-capacitor/RJD2032C1ST1/1572-1628-ND/6596416
TL3305AF160QG	Pushbutton Switch	http://spec_sheets.e-switch.com/specs/P010468.pdf
S-8261	Battery Protection IC	https://www.ablic.com/en/doc/datasheet/battery_protection/S8261_E.pdf
RC0603JRL	Surface-mount Chip Resistor	https://www.yageo.com/upload/media/product/productsearch/datasheet/rchip/PYu-RC_Group_51_RoHS_L_10.pdf
RNCP Series	Surface-mount Chip Resistor	https://www.seielect.com/catalog/sei-rncp.pdf
GRM0335 Series	Surface-mount Ceramic Capacitors	https://search.murata.co.jp/Ceramy/image/img/A01X/G101/ENG/GRM0335C1E120JA01-01.pdf
ECS-.327-8-14	Watch Crystal (32kHz)	https://ecsxtal.com/store/pdf/ECS-3x8.pdf
Ginsco Watch Pin Set	Watch Pin	https://www.amazon.com/Ginsco-6-25mm-Stainless-Spring-Remover/dp/B01C27A45O/ref=sr_1_4?keywords=watch+pins&qid=1576523763&sr=8-4
WOCCI Watch Band 18 mm	Watch Band	https://www.amazon.com/WOCCI-18mm-Watch-Band-Stitching/dp/B01N0R03V8/ref=sr_1_13?keywords=watch+band&qid=1576523702&sr=8-13

Figure 7: Components Used

Figure 7 lists all the electronic and mechanical components used to assemble the final device.

Circuit Schematics

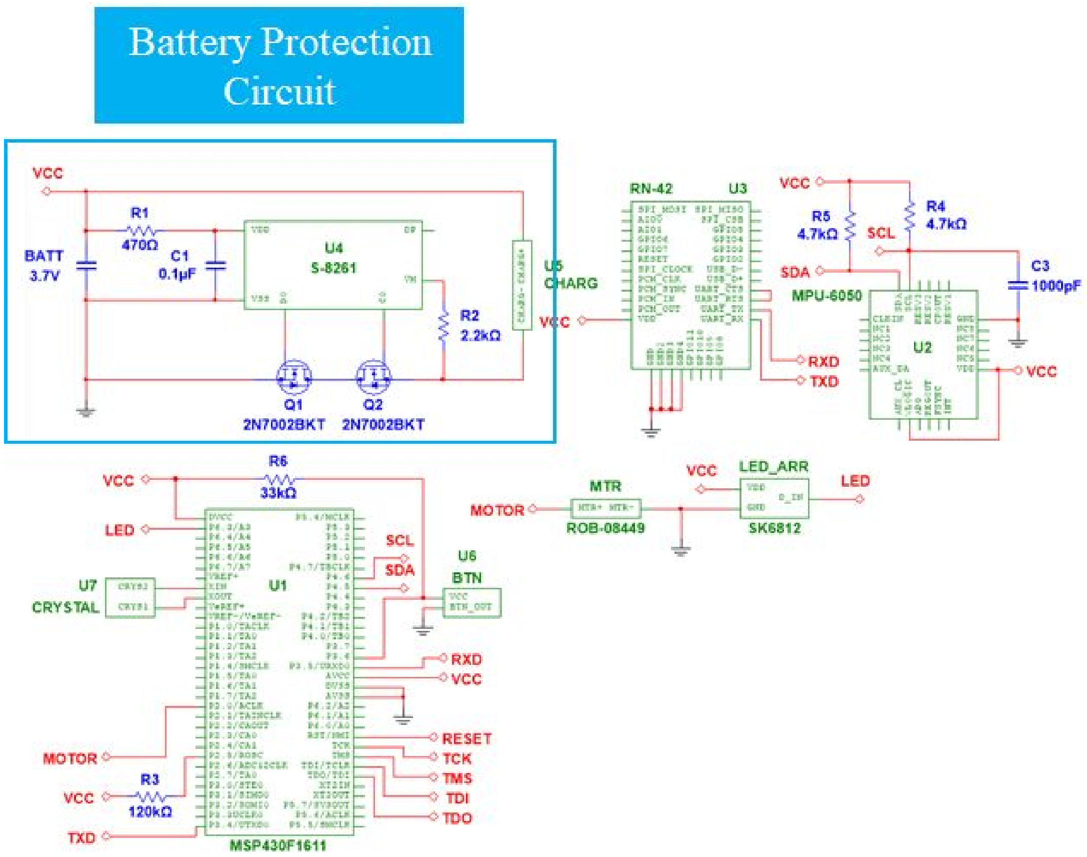


Figure 8: Original Circuit Schematic (Watch PCB)

Figure 8 shows our original schematic for the watch PCB, including each of the planned subsystems interfaced with the MSP430. All components are supplied by VCC power from the battery protection circuit labeled.
The power circuit consists of the charging receiver (U5) connected to the main battery through the battery protection integrated circuit (U4), which regulates the battery's charging and discharging rates based on its capacity. MOSFETs Q1 and Q2 direct current as needed during the two extremes of the battery's use-cases: where it is allowed to charge when empty and discharge when full to prevent under/overvoltage.

The RN-42 Bluetooth module (U3) communicates with the MSP430 (U1) via UART serial interface, where its receiver port RX connects with the MSP430's dedicated transmitter pin and its transmitter port TX connects with the MSP430's dedicated receiver pin, allowing a two-way communication for Bluetooth standard protocol.

The MPU-6050 gyroscope/accelerometer (U2) interfaces with the MSP430 with a data wire SDA and a clock wire SCL. Both lines require pull-up resistors to VCC power (R4 and R5) and the clock requires bypass capacitor C3. The MPU-6050 implements a I2C protocol for data transfer, however, the MSP430F1611 does not have specialized ports for a hardware-based I2C. Thus, we connected the lines to two of our MSP430's general-purpose I/O (GPIO) ports, at which point our custom software would handle the I2C protocol to relay motion data.

The LED ring's only requirement besides power/ground is a single data input line to GPIO using a custom driver protocol to control all 12 LEDs' color at one time. The 32kHz watch crystal (U7) enables the MSP430 to have precise clock speed (and thus general timekeeping) by connecting to its XIN/XOUT pins which are dedicated for crystal oscillators. Since the LED driver's protocol requires at least 6MHz timing for communication, the MSP430's clock frequency had to be increased from its default 5MHz to 8.4MHz via the pull-up resistor R3.

The pushbutton (U6) is enabled as an active low, where by default the VCC supply is input to the GPIO button input pin and consequently grounded whenever the button switch is pressed/closed.

The JTAG interface needed to flash our software to the MSP430, consisting of reset/test clock/test mode select/test data in/data out pins (RST/TCK/TMS/TDI/TDO), was initially planned as separate through-hole connections that would be connected to the debugging interface with separate jumper wires.

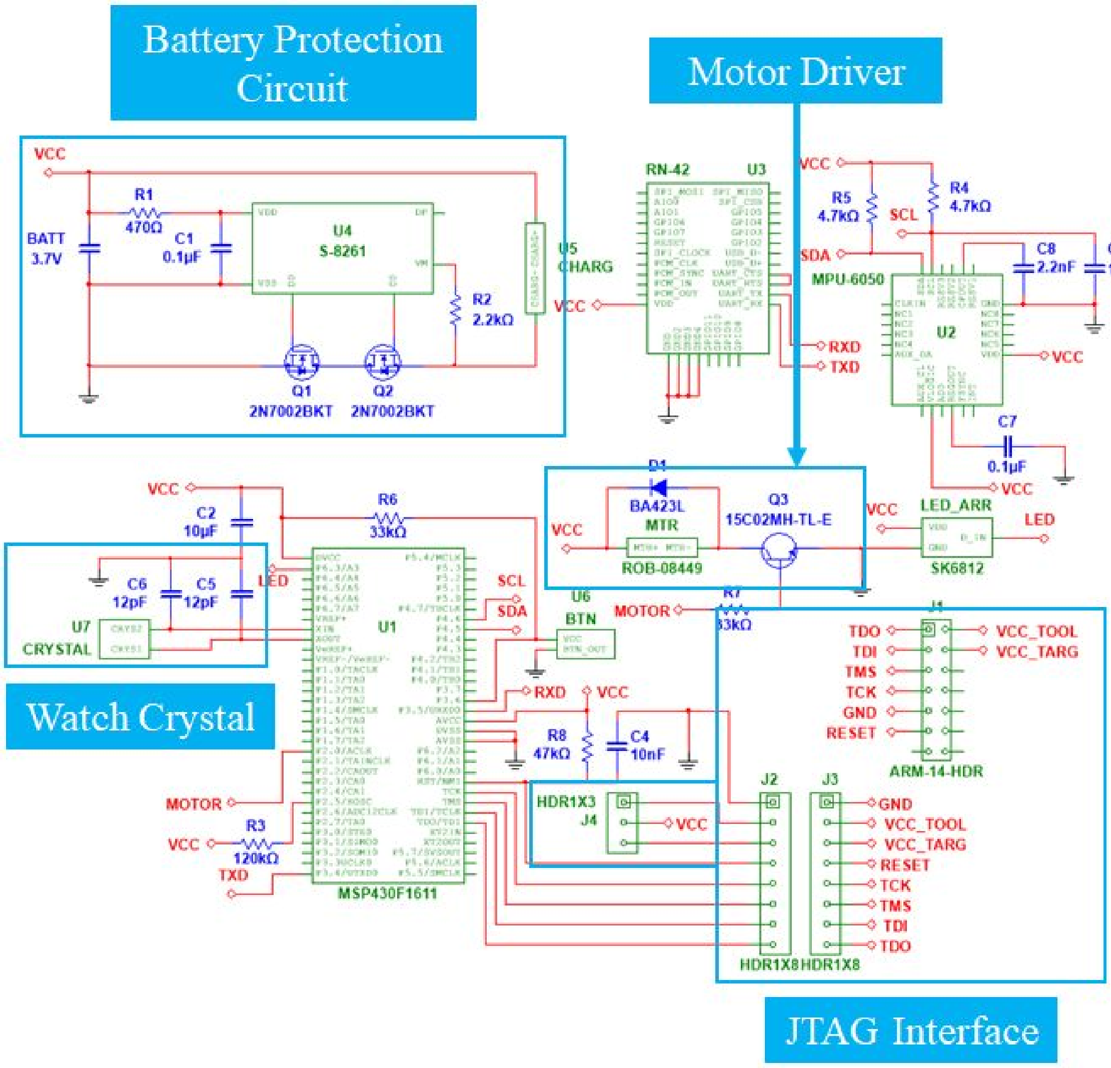


Figure 9: Final Circuit Schematic (Watch and JTAG Helper PCB)

Figure 9 shows the finalized schematic for the watch PCB and auxiliary components, including several changes to the MSP430 (U1), MPU-6050 (U2), watch crystal (U7), vibration motor, and JTAG interface. The battery protection (U4) and power circuit, as well as the Bluetooth module (U3), were unchanged from the original design, as their functionalities had been verified.

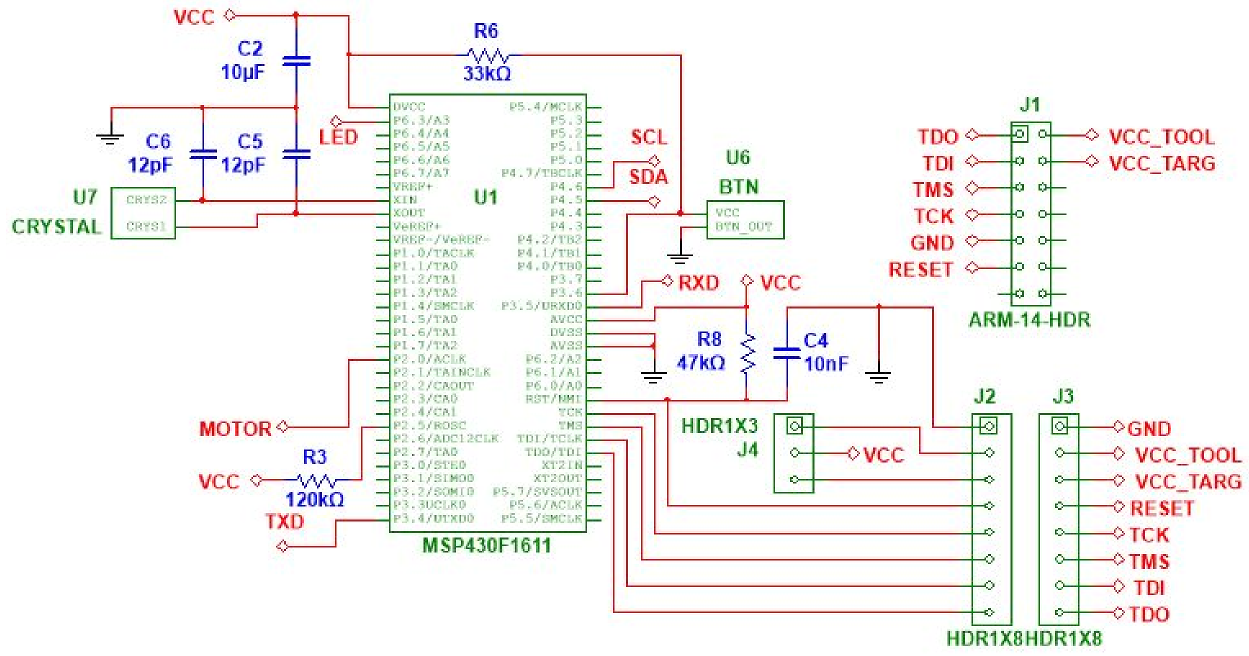


Figure 10: Final MSP430 and JTAG Interface Schematic

Figure 10 focuses on the changes made to the MSP430 and its JTAG interface. Both the MSP430 and watch crystal (U7) had added bypass capacitors C2, C5, and C6 in order to be recognized by the debugging JTAG interface. An RC circuit (R8 and C4) was also added to the reset input for safety and reliability of booting.

The JTAG interface on the PCB was updated to be a right-angle horizontal 8-pin connector (J2) for a low profile and easy connection. In order to transition the JTAG debugger's standard 14-pin connector to this port, an external helper board was created to map a full 14-pin JTAG connector (J3) to a simplified 8-pin output (J2). An auxiliary 1x8 pin ribbon cable was used to connect this system to the watch port.

The updated JTAG interface now also included separate VCCs for both tool and target supplied power. These modes were toggled by a 3-pin connector (J4) on the watch PCB and a shunt to short either the device battery or the debugger-supplied PC power to the global VCC supply. This made it easier to reconfigure the watch for either independent battery powered operation or wired debugger power operation for testing.

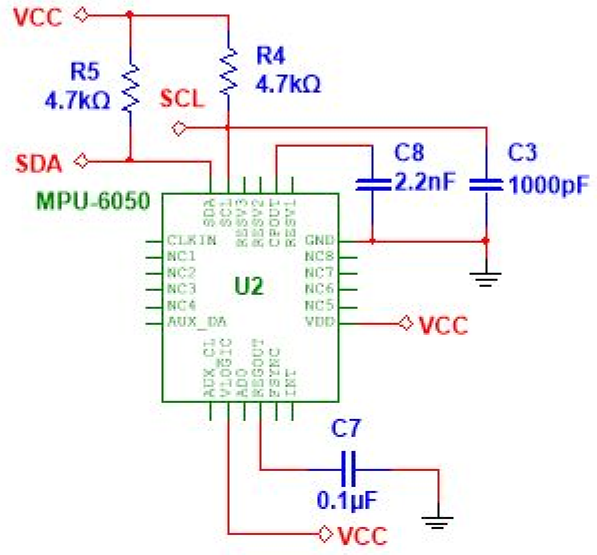


Figure 11: Final MPU-6050 Schematic

Figure 11 focuses on the changes made to the MPU-6050 gyroscope/accelerometer. In order for motion data to be effectively calibrated and sent, we had to add bypass capacitors for the regulator filter and the on-board charge pump necessary for its oscillators (C7 and C8).

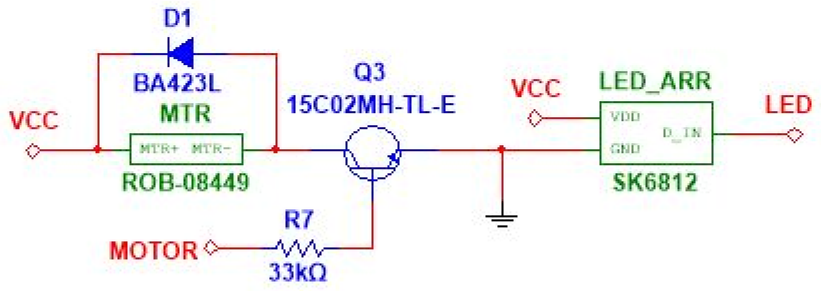


Figure 12: Final Motor and LED Driver Schematic

Figure 12 focuses on the changes made to the vibration motor. The motor interface was updated to a BJT driver wherein an enable from the designated GPIO motor pin on the MSP430 allowed the motor to be powered by the global VCC supply. A flyback diode D1 was added in parallel to the motor to prevent the damaging effects of sudden pulses from the motor's inductive load upon sudden throttling of current.

Board Layout

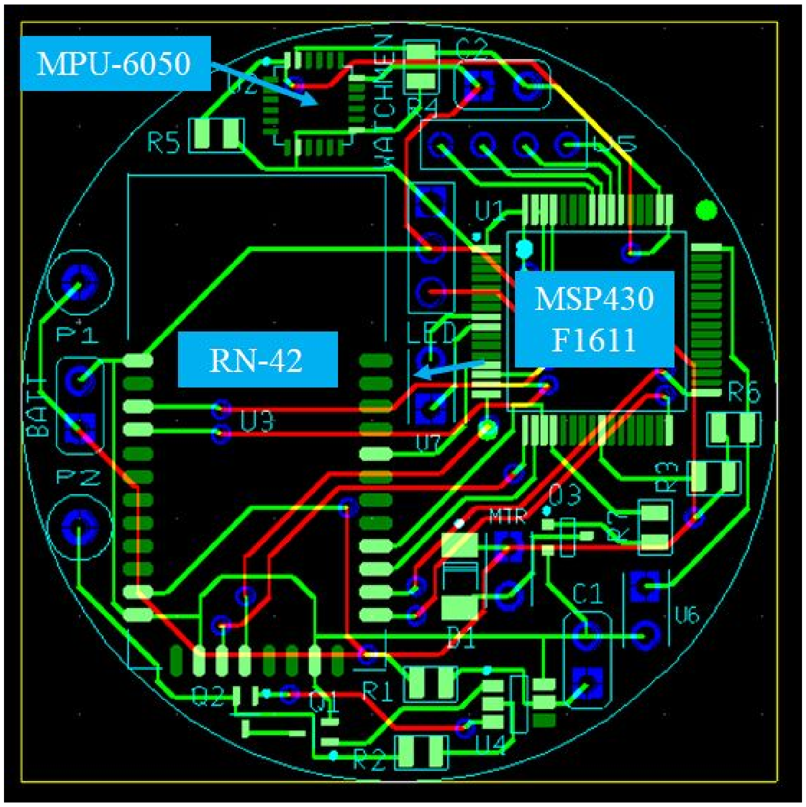


Figure 13: Original Watch PCB Layout

Figure 13 shows the board layout for the originally planned schematic. The coin battery and charging receiver coil were larger discrete components that were planned to be placed under the PCB to retain the 40mm watch diameter. Thus, their footprints were designed as through-hole groups (BATT and P1/P2) so they could be connected from under via wires. The vibration motor (MTR) also followed this convention, so that the user could feel vibration feedback well through the bottom of the watch.

The LED ring and pushbutton (U6) also required similar through-hole/wire connections, as they had to be placed on top of the PCB. The LED ring would rest parallel to the PCB facing upwards and the pushbutton would be fixed to the side of the display face housing.

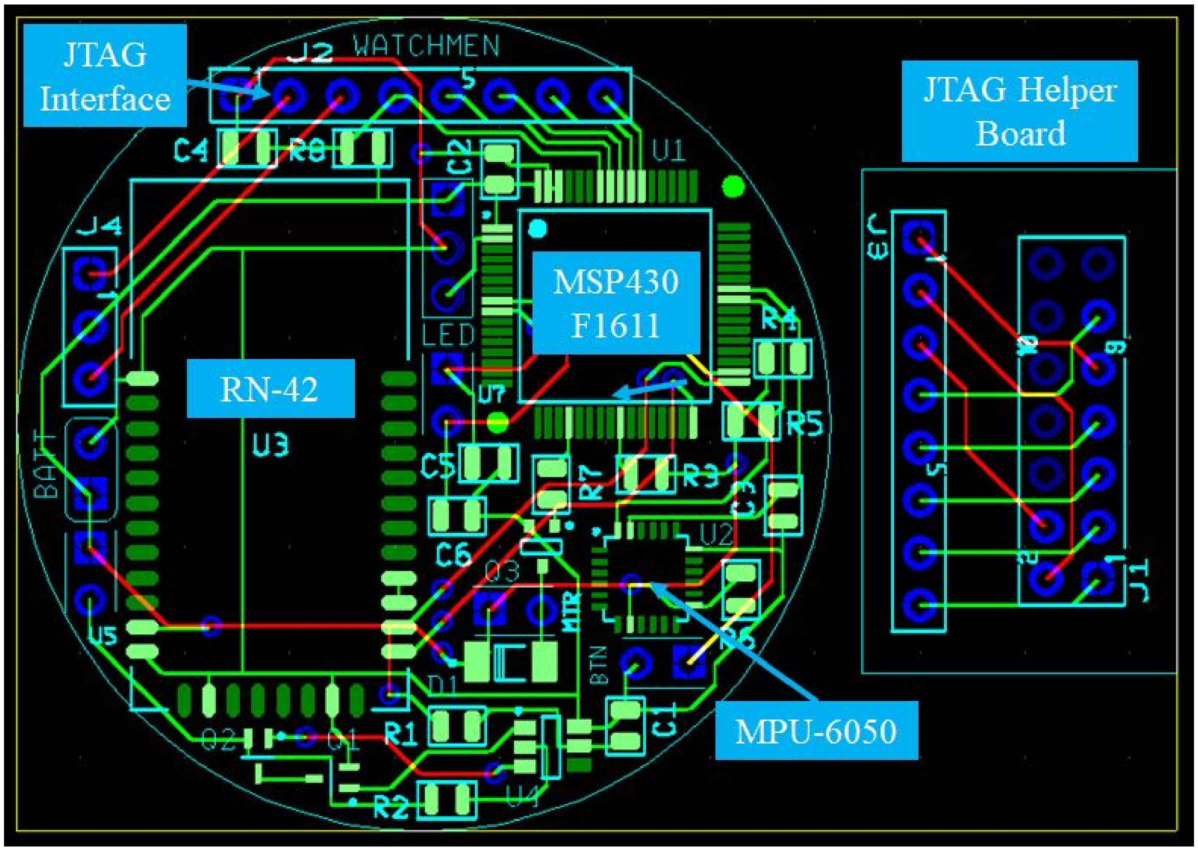


Figure 14: Final Watch and JTAG Helper PCB Layout

Figure 14 shows the finalized layout for both the watch PCB and the JTAG helper board. All external components retained their through-hole connections for flexible placement in the final watch housing. The added bypass capacitors were placed as close as possible to their corresponding devices, and due to the increase in required components, the capacitors were downsized to the same 0603 footprint as the resistors to reserve space.

The 1x8 JTAG through-holes (J2) on the top edge of the board would have a matching right-angle connector soldered to it that would act as a connector port visible from the side of the watch body. This port would lead to the corresponding connector J3 that remaps to the standard 14-pin JTAG connector J1 on the helper board. The two boards were separated and machined out at 3W before mounting.

Not pictured in the final PCB layout are the bypass capacitors for the MPU-6050 regulator filter and charge pump, which were forgotten and included later in assembly.

Design Decisions and Tradeoffs

Many design choices and additional functionalities were at odds with our desire to keep the watch compact and efficient for aesthetic and user appeal. The following are significant design decisions and tradeoffs made while progressing from the original design to the final design as described above:

- Using the MSP430F1611 microcontroller: this MCU was chosen for its overall popularity and reliability for low-power mobile/wearable devices, although in using it we sacrificed the benefit of a simple I2C interface with the MPU-6050 gyroscope/accelerometer. Consequently, we had to spend extra development on a software-defined I2C protocol for motion sensing.
- 8-pin JTAG port with helper board: The slimmed port on the side of the watch with an external helper board was an acceptable compromise between two extremes: mounting a fully featured 14-pin connector (which would have made the watch unpleasantly bulky) or eliminating an accessible JTAG port altogether (which would have prevented us from being able to make future software changes/fixes).
- Tool/target VCC switching: Like the JTAG port, the shunt toggle between local and debugger power allowed easier reconfiguration for future testing and improvements, at the cost of a slightly higher watch profile and PCB density.
- Wired LED ring and pushbutton connection: Although direct solder connections or PCB mounting would have allowed a lower profile and compact assembly, we opted for a slightly larger yet modular assembly process that allowed us to experiment with different configurations/orientations of the upper chassis.
- Reset input RC circuit: The circuit configuration for the reset input was determined to be good practice for maintaining a stable system, at the cost of taking up more valuable PCB space.

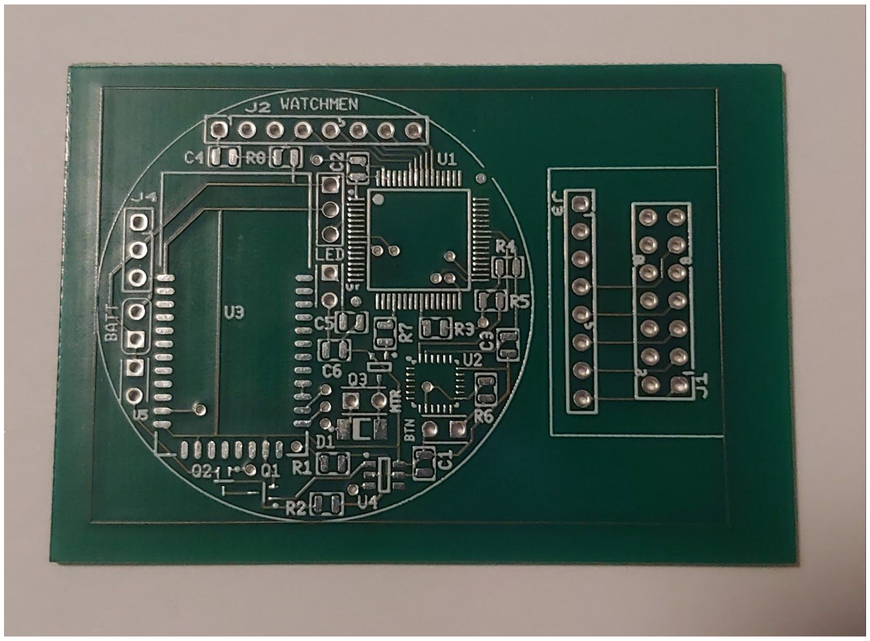


Figure 15: Final Watch and JTAG Helper PCB Before Cutting/Mounting

Figure 15 shows our finalized PCB with JTAG helper board to the right of it.

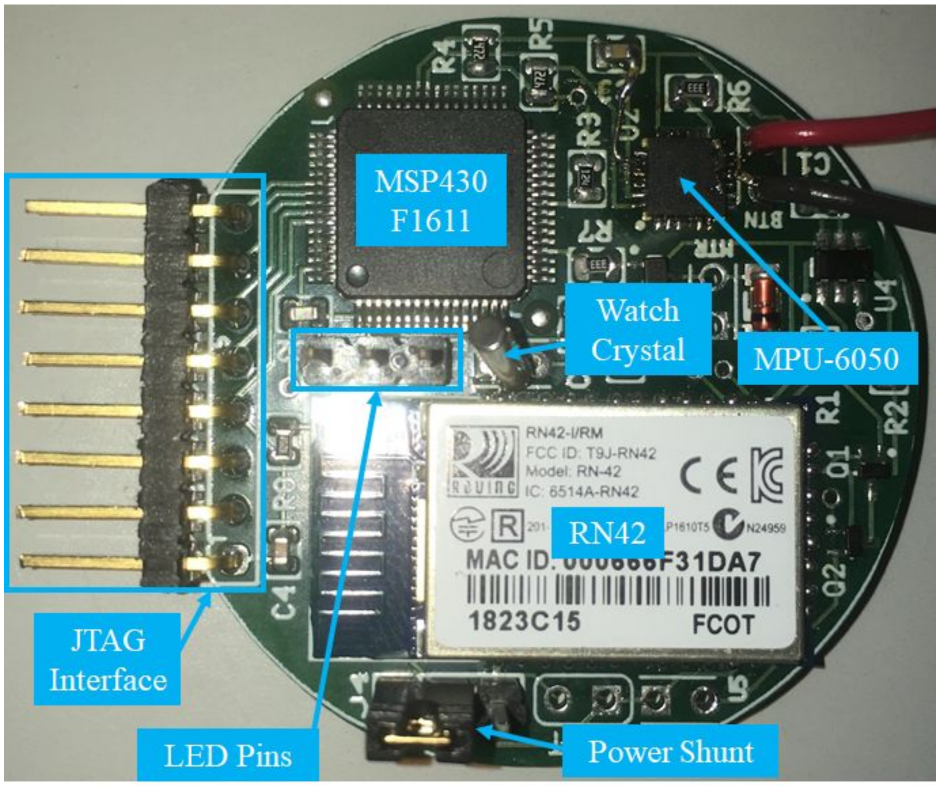


Figure 16: Final Watch PCB with Onboard Parts Mounted

Figure 16 shows our finalized PCB with most of our onboard parts mounted.

Watch Header Board Testing

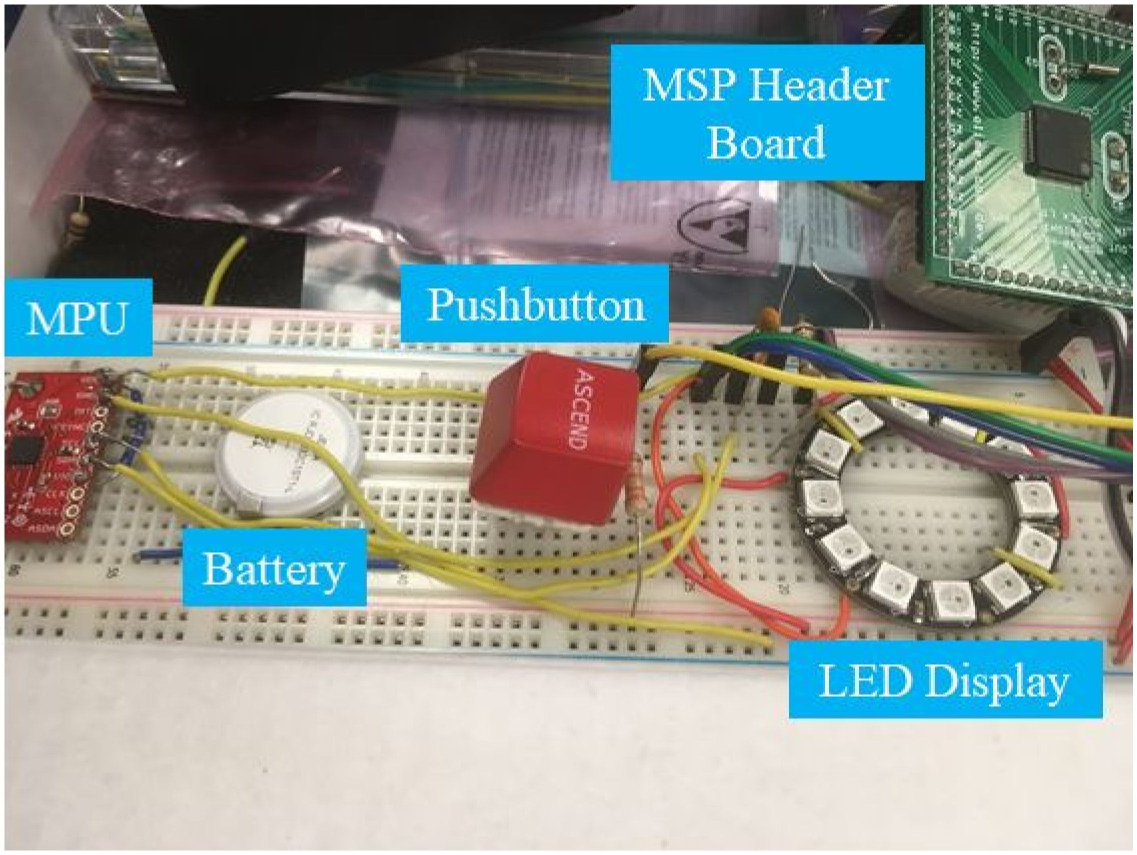


Figure 17: Header Board Testing Setup

The above Figure 17 shows how preliminary testing was done for the project. The header boards from Debugging Components Figure 6 were used to construct a prototype system. A generic pushbutton was used at the time because the final ordered pushbutton had not come in.

Watch Chassis Modeling

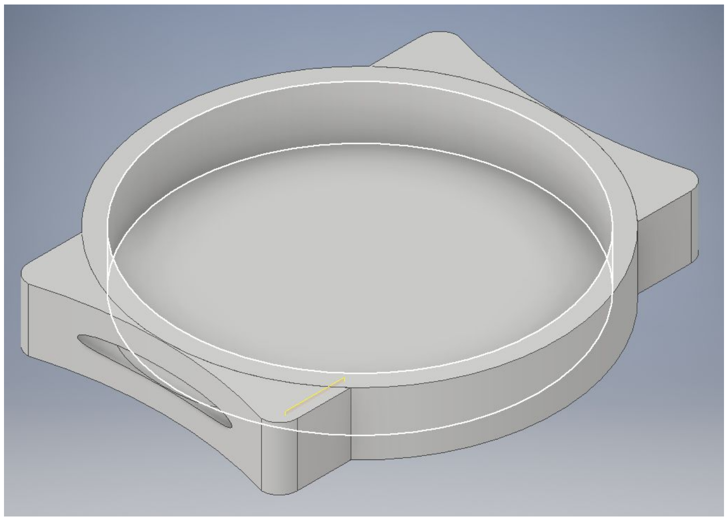


Figure 18: Angle View Original Bottom Chassis

The above Figure 18 shows the original bottom chassis that was developed. The original idea was to have the watch PCB fit into the circular cavity with all the external components such as the battery and vibration motor under it.

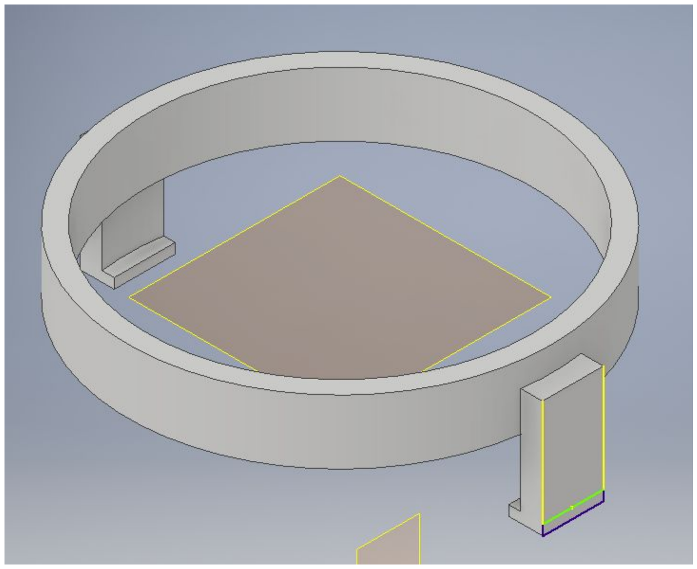


Figure 19: Angle View Original Top Chassis

The above Figure 19 shows the original top chassis that was developed. The original idea was to have the LED display fit into the top chassis by friction and having the clips on the side attached to the bottom chassis.

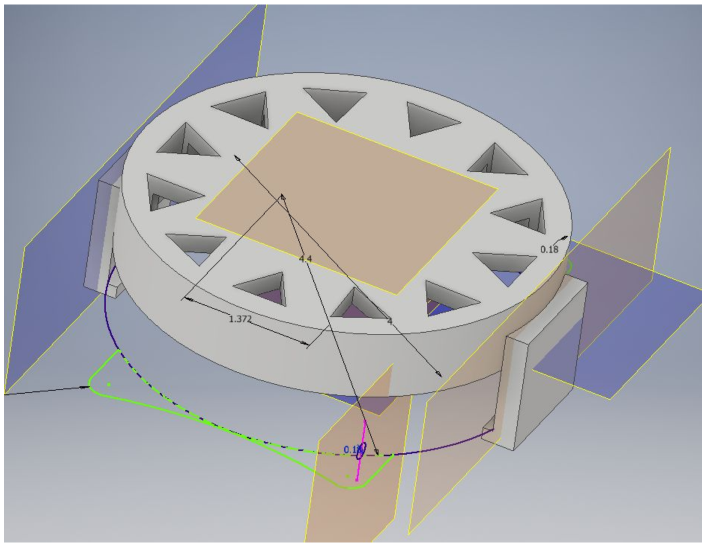


Figure 20: Angle View Modified Top Chassis

Figure 20 above shows a modified version of the top chassis in Inventor [21]. This version includes a fully enclosed housing for the PCB and circuitry, cut outs for the LED display and a larger clip length to account for tolerance. These cutouts for the LEDs allows light to shine through and users of the watch to accurately see the LEDs without adjacent LED lights bleeding over.

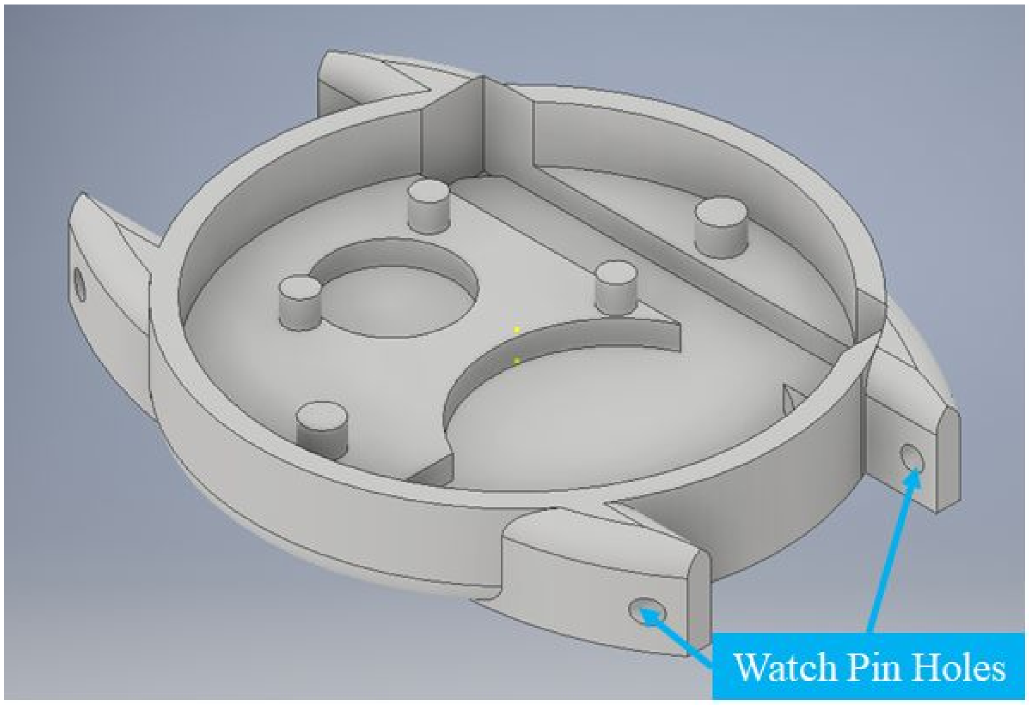


Figure 21: Angle View Final Bottom Chassis

Figure 21 above shows the finalized bottom chassis. There are watch pin holes to account for the watch band and the 20mm compression watch pin.

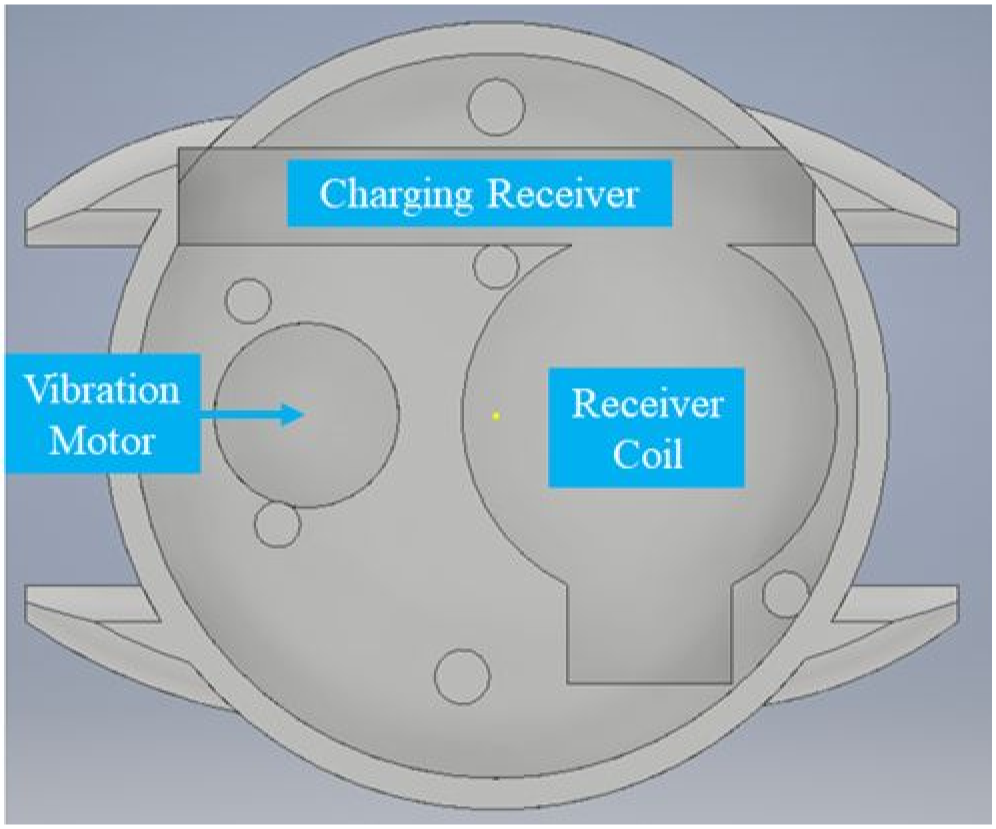


Figure 22: Top View Final Bottom Chassis

There are standoff columns for the PCB to rest on top of and specialized cutouts for the vibration motor, charging receiver, and charging coil to rest in. This design change was made to ensure that there were no loose components. This can be seen in Figure 22.

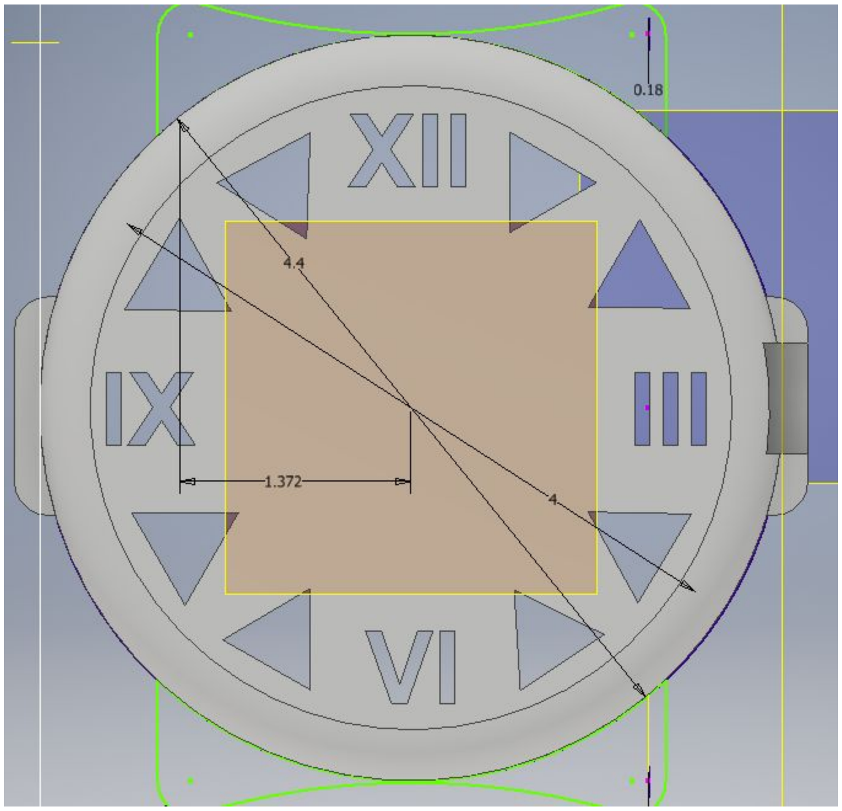


Figure 23: Top View Final Top Chassis

Figure 23 shows the final top chassis. There are cutouts indicating the major time values located at 3, 6, 9, and 12 o'clock.

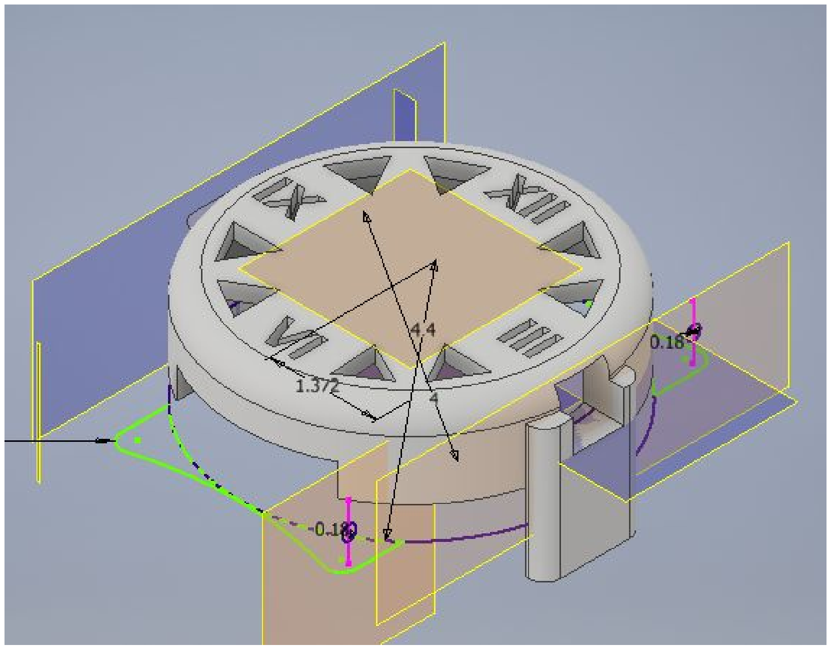


Figure 24: Angle View Final Top Chassis

Figure 24 shows an angled view of the final top chassis. It was also made slightly taller in height to account for the components and wires sticking out from the PCB. There is also now an opening on the side for where the pushbutton is supposed to be secured.

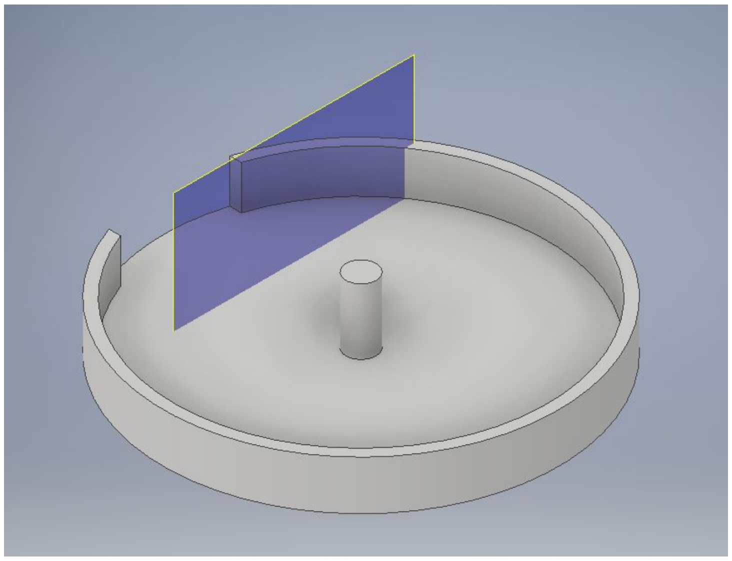


Figure 25: Angle View Final Bottom Charger Chassis

Figure 25 above shows the final bottom charger chassis. This is meant to house the transceiver coil. There is a column in the middle that the transceiver coil is meant to go around and an opening to the side that lets the coil wires in.

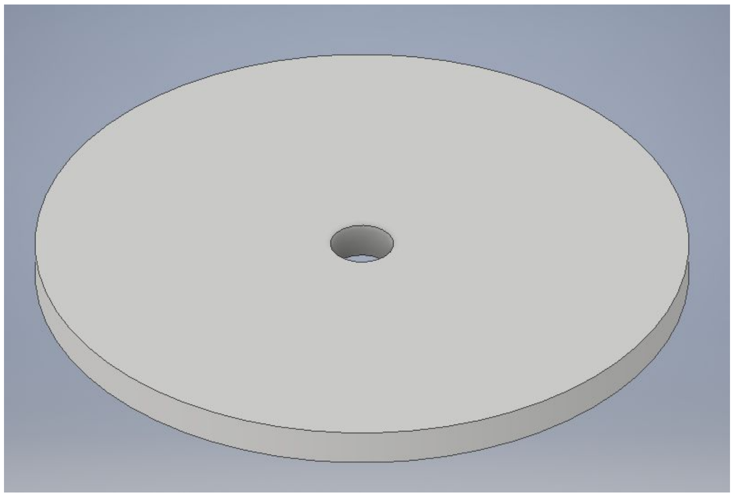
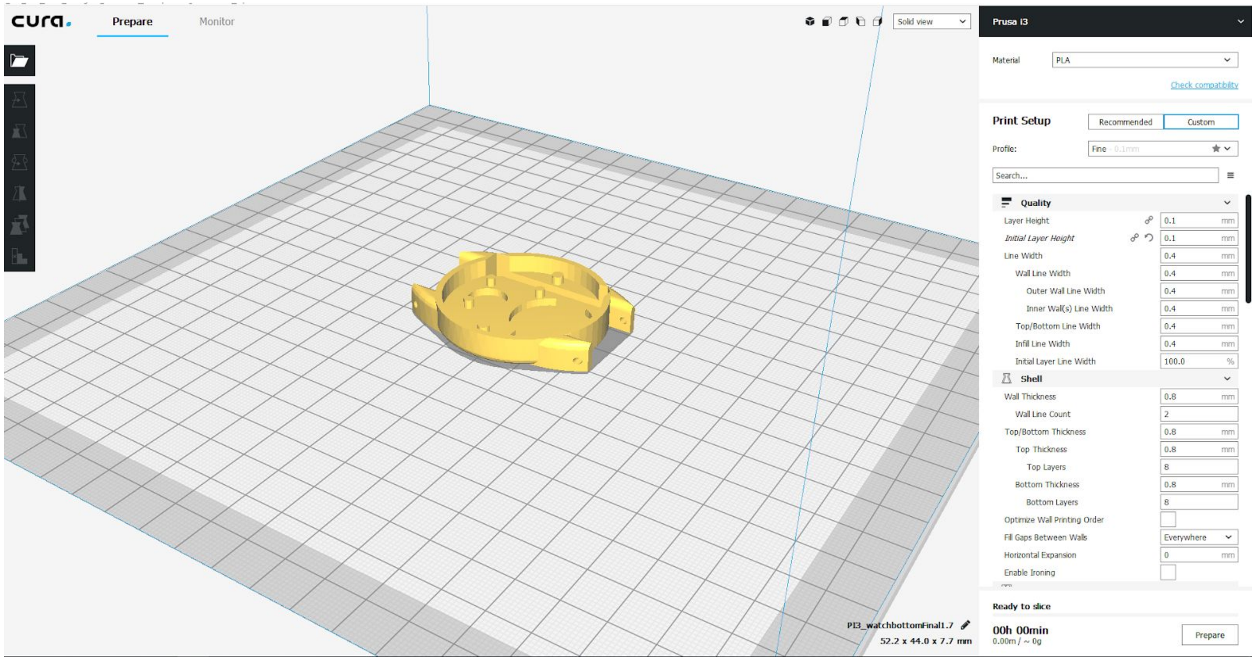


Figure 26: Angle View Final Top Charger Chassis

The above Figure 26 shows an angled view of the final top charger chassis. There is a hole in the middle that is meant to clip into the column of the bottom chassis.



Prusa i3

Material: PLA

Check compatibility

Print Setup — Recommended | Custom

Profile: Fine - 0.1mm

Search...

Quality

Layer Height	0.1 mm
Initial Layer Height	0.1 mm
Line Width	0.4 mm
Wall Line Width	0.4 mm
Outer Wall Line Width	0.4 mm
Inner Wall(s) Line Width	0.4 mm
Top/Bottom Line Width	0.4 mm
Infill Line Width	0.4 mm
Initial Layer Line Width	100.0 %

Shell

Wall Thickness	0.8 mm
Wall Line Count	2
Top/Bottom Thickness	0.8 mm
Top Thickness	0.8 mm
Top Layers	8
Bottom Thickness	0.8 mm
Bottom Layers	8
Optimize Wall Printing Order	☐
Fill Gaps Between Walls	Everywhere
Horizontal Expansion	0 mm
Enable Ironing	☐

Ready to slice

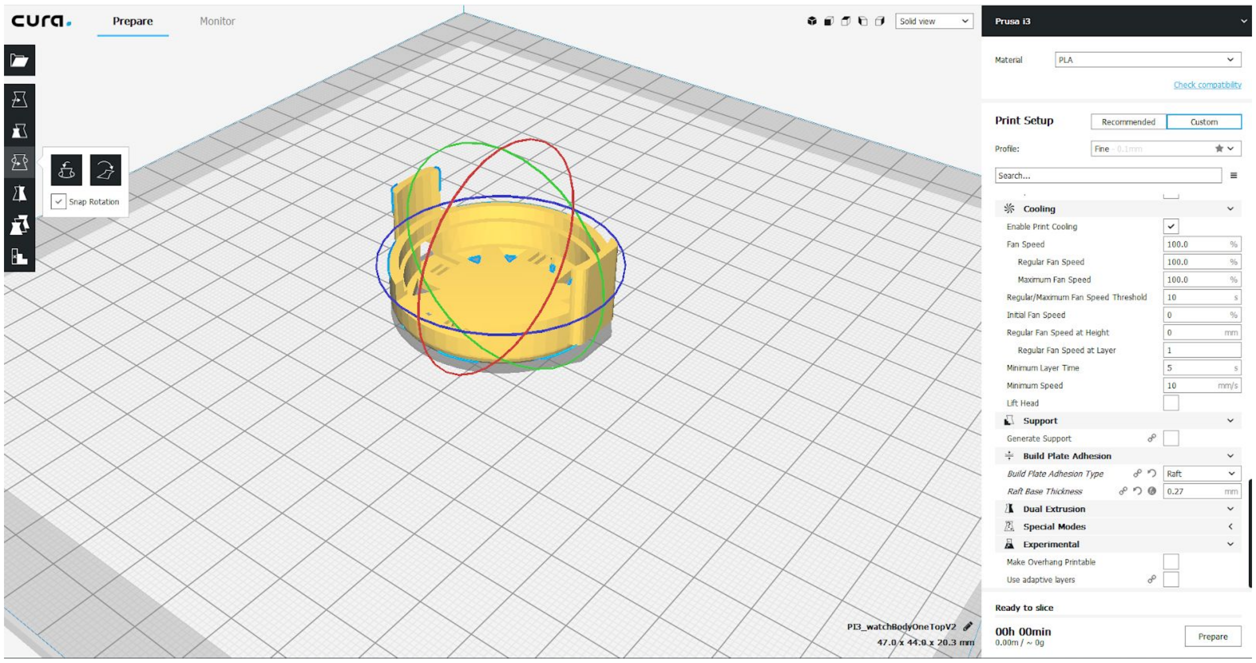
00h 00min
0.00m / ~ 0g

Prepare

PI3_watchbottomFinal1.7
52.2 x 44.0 x 7.7 mm

Figure 27: Final Bottom Chassis in Cura

The above Figure 27 shows the final bottom chassis in Cura.



Prusa i3

Material: PLA

Check compatibility

Print Setup — Recommended | Custom

Profile: Fine - 0.1mm

Search...

Cooling

Enable Print Cooling	✓
Fan Speed	100.0 %
Regular Fan Speed	100.0 %
Maximum Fan Speed	100.0 %
Regular/Maximum Fan Speed Threshold	10 s
Initial Fan Speed	0 %
Regular Fan Speed at Height	0 mm
Regular Fan Speed at Layer	1
Minimum Layer Time	5 s
Minimum Speed	10 mm/s
Lift Head	☐

Support

Generate Support	☐

Build Plate Adhesion

Build Plate Adhesion Type	Raft
Raft Base Thickness	0.27 mm

Dual Extrusion

Special Modes

Experimental

Make Overhang Printable	☐
Use adaptive layers	☐

Ready to slice

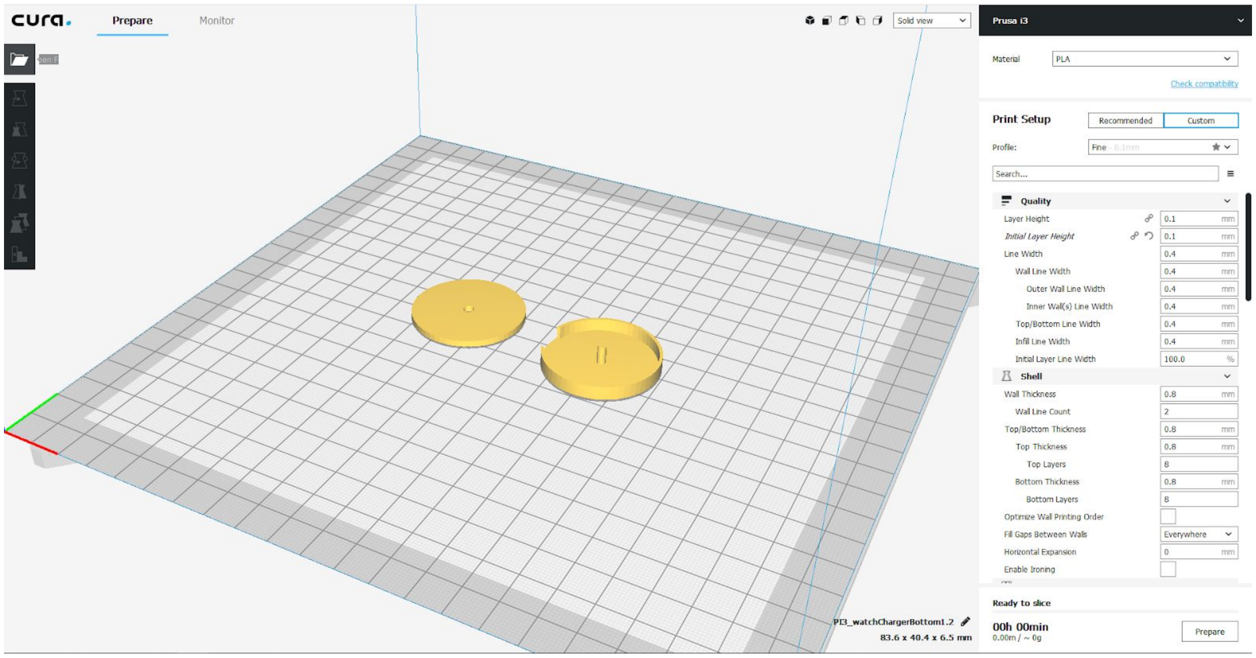
00h 00min
0.00m / ~ 0g

Prepare

PI3_watchBodyOneTopV2
47.0 x 44.0 x 20.3 mm

Figure 28: Final Top Chassis in Cura

The above Figure 28 shows the final top chassis in Cura.



Material: PLA

Check compatibility

Print Setup: Recommended | Custom

Profile: Fine - 0.1mm

Search...

Quality

Layer Height	0.1 mm
Initial Layer Height	0.1 mm
Line Width	0.4 mm
Wall Line Width	0.4 mm
Outer Wall Line Width	0.4 mm
Inner Wall(s) Line Width	0.4 mm
Top/Bottom Line Width	0.4 mm
Infill Line Width	0.4 mm
Initial Layer Line Width	100.0 %

Shell

Wall Thickness	0.8 mm
Wall Line Count	2
Top/Bottom Thickness	0.8 mm
Top Thickness	0.8 mm
Top Layers	8
Bottom Thickness	0.8 mm
Bottom Layers	8
Optimize Wall Printing Order	☐
Fill Gaps Between Walls	Everywhere
Horizontal Expansion	0 mm
Enable Ironing	☐

Ready to slice

00h 00min
0.00m / ~ 0g

Prepare

PI3_watchChargerBottom1.2
83.6 x 40.4 x 6.5 mm

Figure 29: Final Charger Chassis in Cura

The above Figure 29 shows the final charger top and bottom chassis in Cura.

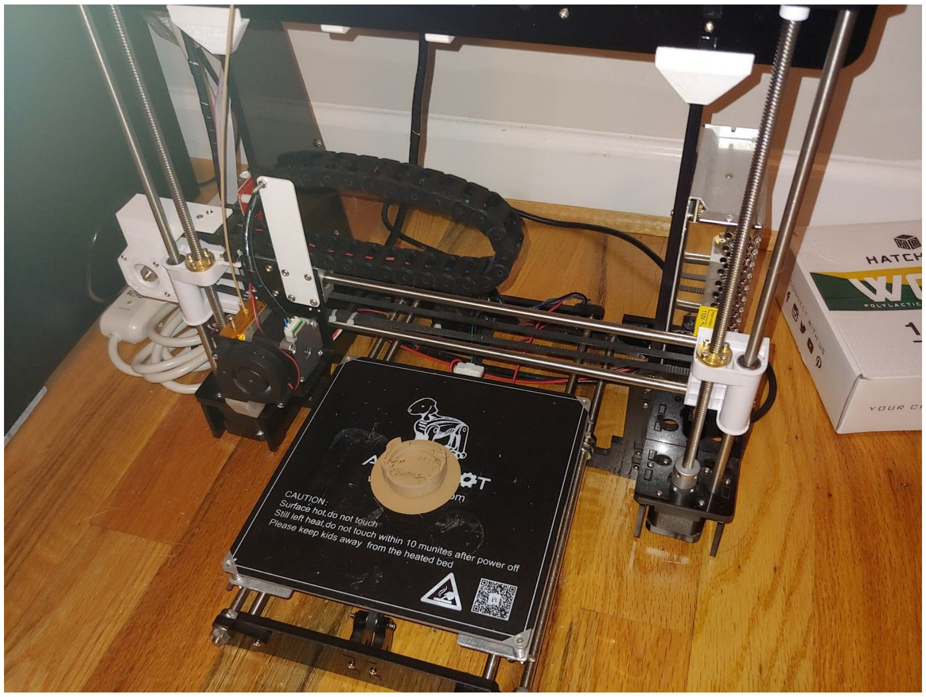

CAUTION:
Surface hot,do not touch
Still left heat,do not touch within 10 munites after power off
Please keep kids away from the heated bed
HATCH

Figure 30: Printing Bottom Chassis

The above Figure 30 show the final top chassis being printed on an ANet A8 [?]. All of the other chassis were printed in the same way.

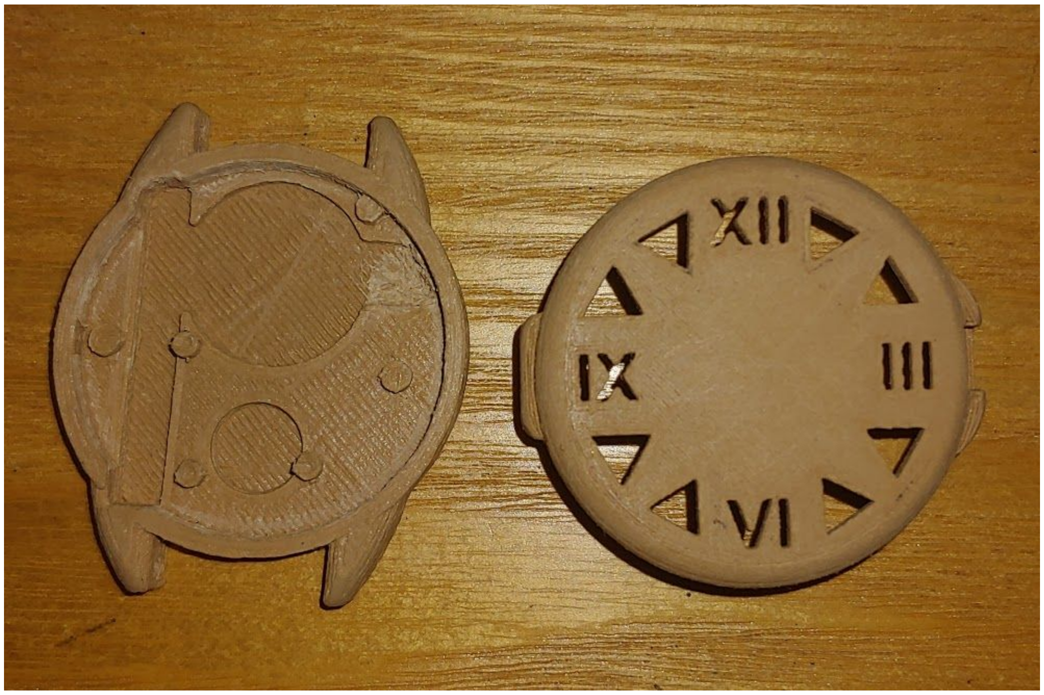


Figure 31: Bottom and Top Chassis

The above Figure 31 shows the final top and bottom chassis after being printed.

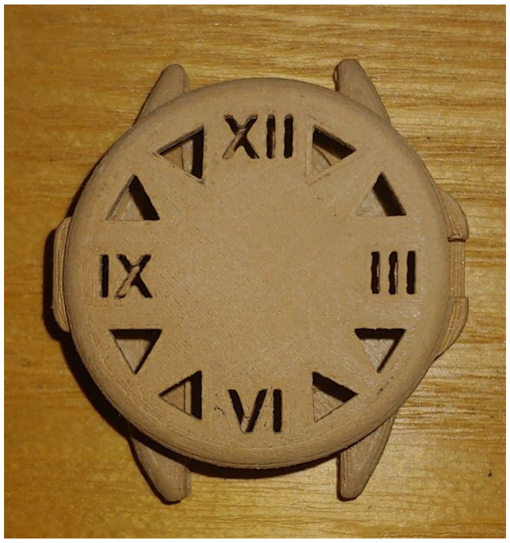


Figure 32: Bottom and Top Chassis Together

The above Figure 32 shows the final top and bottom chassis clipped together.

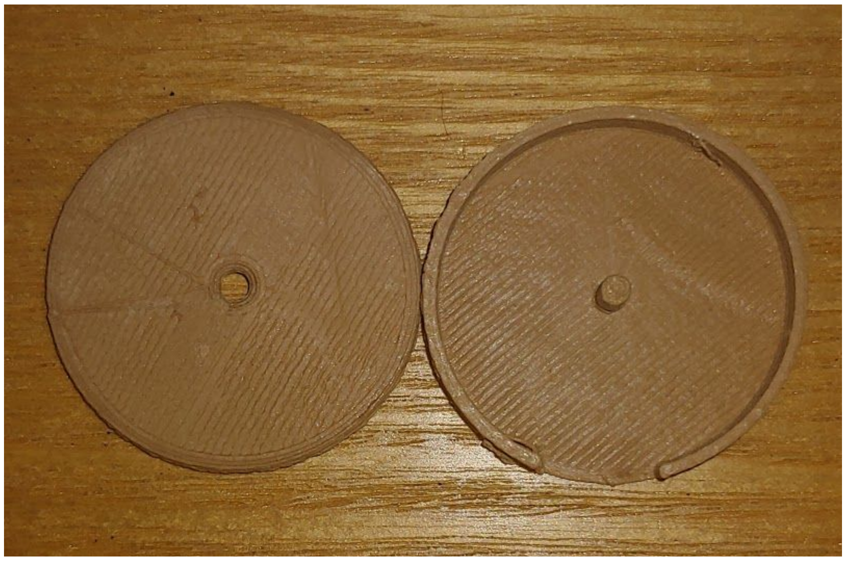


Figure 33: Bottom and Top Charger Chassis

The above Figure 33 show the final top and bottom charger chassis after being printed.

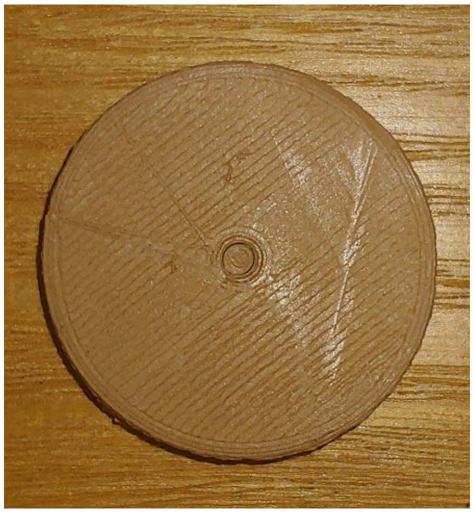


Figure 34: Bottom and Top Charger Chassis Together

The above Figure 34 shows the final top and bottom charger chassis clipped together.

MSP430 and AutoIt Code

```c
void SendData(void)
{
    unsigned char i, j, k;
    for (i = 0; i < 12; i++)
    {
        for (j = 0; j < 3; j++)
        {
            for (k = 7; k < 8; k--)
            { /*k<8 because (unsigned 0) - (unsigned 1) equals INT_M_ax; */
                if ((led_data[j + i * 3] >> k) & 0x01)
                {
                    TURN_ON_LED;
                    TURN_ON_LED;
                    TURN_OFF_LED;
                }
                else
                {
                    TURN_ON_LED;
                    TURN_OFF_LED;
                }
            }
        }
    }
}
```

Figure 35: LED SendData

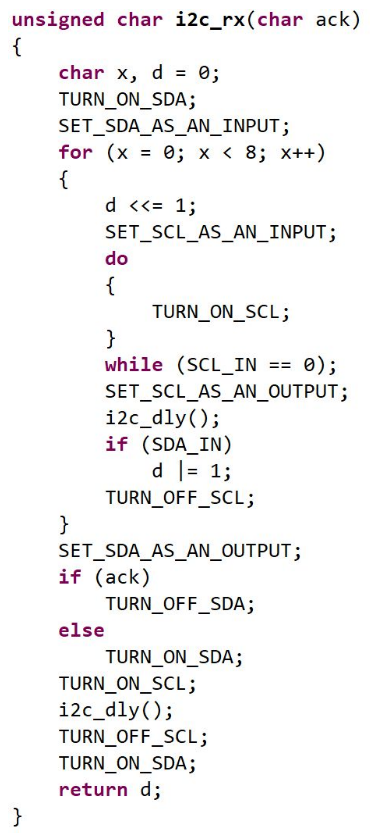
The SendData function takes the stored array of LED data and transmits it to the LED display. The way this LED ring functions, it interprets a very short ‘high’ voltage as a ‘zero’ and any ‘high’ voltage longer than that as a ‘one’. It was tricky to get the timing right for this, and we had to add an external resistor so that we could run our main clock at 8 MHz. It takes a minimum of four main clock cycles to toggle a pin high and low again, and 8 MHz is just fast enough that those four clock cycles are still seen as a ‘zero’.
The part with two ‘TURN_ON_LED’ lines in a row is intended to stretch that ‘high’ voltage out a little longer, so it is read as a ‘one’. The part with only one ‘TURN_ON_LED’ is read as a ‘zero’.

```c
void ReadGyro()
{
    uint8_t raw_data[6];  /* x/y/z gyro register data stored here */
    ReadBytes(MPU6050_ADDRESS, GYRO_XOUT_H, 6, &raw_data[0]); /* Read the two raw data registers sequentially into data array */
    gx = ((int16_t) raw_data[0]) << 8 | raw_data[1]; /* Turn the MSB and LSB into a 16-bit value */
    gy = ((int16_t) raw_data[2]) << 8 | raw_data[3]; /* Turn the MSB and LSB into a 16-bit value */
    gz = ((int16_t) raw_data[4]) << 8 | raw_data[5]; /* Turn the MSB and LSB into a 16-bit value */
}

void ReadAccel()
{
    uint8_t raw_data[6];  /* x/y/z gyro register data stored here */
    ReadBytes(MPU6050_ADDRESS, ACCEL_XOUT_H, 6, &raw_data[0]); /* Read the two raw data registers sequentially into data array */
    ax = ((int16_t) raw_data[0]) << 8 | raw_data[1]; /* Turn the MSB and LSB into a 16-bit value */
    ay = ((int16_t) raw_data[2]) << 8 | raw_data[3]; /* Turn the MSB and LSB into a 16-bit value */
    az = ((int16_t) raw_data[4]) << 8 | raw_data[5]; /* Turn the MSB and LSB into a 16-bit value */
}
```

Figure 36: ReadGyro and ReadAccel

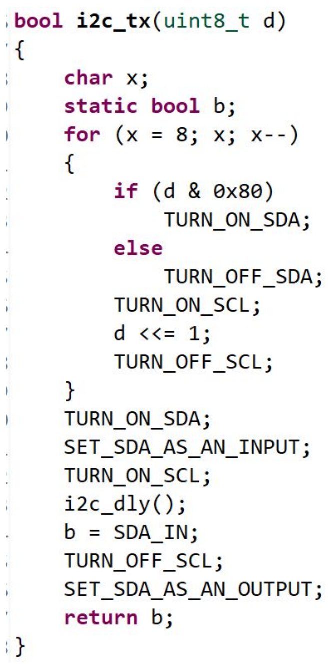
The functions ReadGyro and ReadAccel both use the function ‘ReadBytes’ to fetch data from sequential registers of the MPU6050 using software implemented I2C.



```c
unsigned char i2c_rx(char ack)
{
    char x, d = 0;
    TURN_ON_SDA;
    SET_SDA_AS_AN_INPUT;
    for (x = 0; x < 8; x++)
    {
        d <<= 1;
        SET_SCL_AS_AN_INPUT;
        do
        {
            TURN_ON_SCL;
        }
        while (SCL_IN == 0);
        SET_SCL_AS_AN_OUTPUT;
        i2c_dly();
        if (SDA_IN)
            d |= 1;
        TURN_OFF_SCL;
    }
    SET_SDA_AS_AN_OUTPUT;
    if (ack)
        TURN_OFF_SDA;
    else
        TURN_ON_SDA;
    TURN_ON_SCL;
    i2c_dly();
    TURN_OFF_SCL;
    TURN_ON_SDA;
    return d;
}
```

Figure 37: i2c_rx()

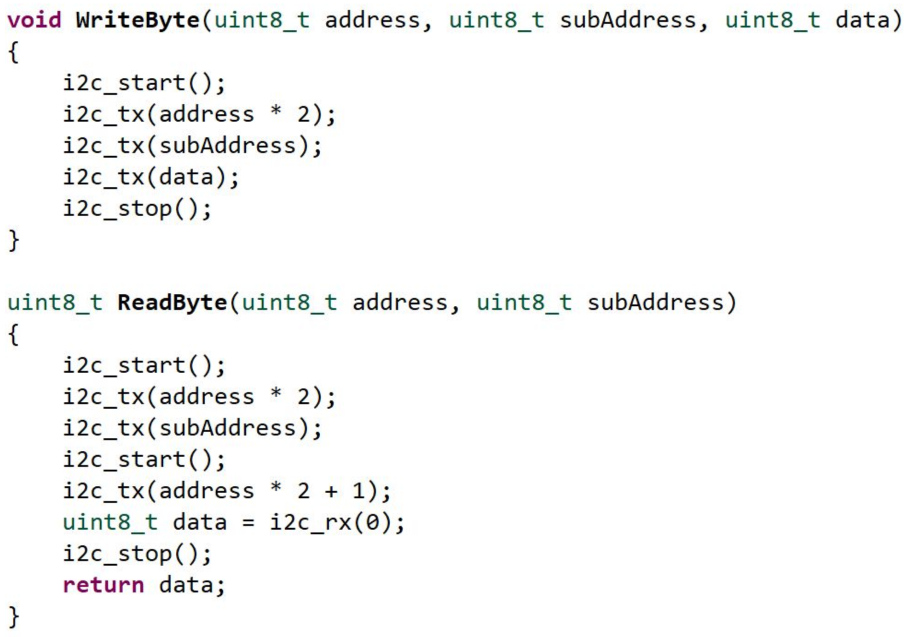
Figure 37: i2c_rx() is used to read a byte coming in over I2C.



```
bool i2c_tx(uint8_t d)
{
    char x;
    static bool b;
    for (x = 8; x; x--)
    {
        if (d & 0x80)
            TURN_ON_SDA;
        else
            TURN_OFF_SDA;
        TURN_ON_SCL;
        d <<= 1;
        TURN_OFF_SCL;
    }
    TURN_ON_SDA;
    SET_SDA_AS_AN_INPUT;
    TURN_ON_SCL;
    i2c_dly();
    b = SDA_IN;
    TURN_OFF_SCL;
    SET_SDA_AS_AN_OUTPUT;
    return b;
}
```

Figure 38: i2c_tx()

i2c_tx is used to send a byte out over i2C.

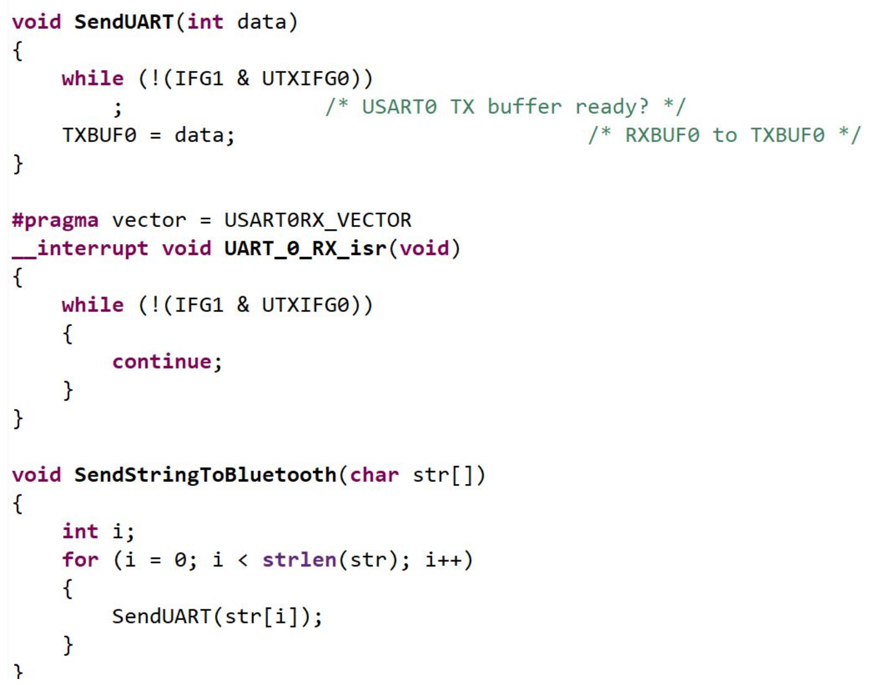


```
void WriteByte(uint8_t address, uint8_t subAddress, uint8_t data)
{
    i2c_start();
    i2c_tx(address * 2);
    i2c_tx(subAddress);
    i2c_tx(data);
    i2c_stop();
}

uint8_t ReadByte(uint8_t address, uint8_t subAddress)
{
    i2c_start();
    i2c_tx(address * 2);
    i2c_tx(subAddress);
    i2c_start();
    i2c_tx(address * 2 + 1);
    uint8_t data = i2c_rx(0);
    i2c_stop();
    return data;
}
```

Figure 39: WriteByte() and ReadByte()

WriteByte and ReadByte utilize the previously mentioned functions in order to read or write bytes to or from specified registers on board the MPU6050 sensor.



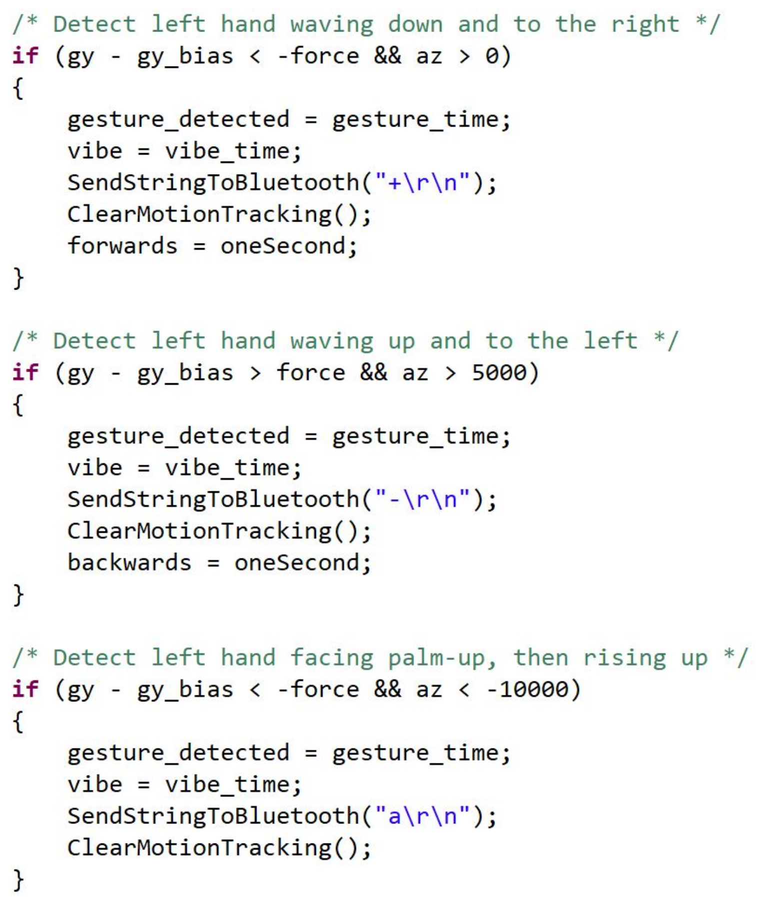
```
void SendUART(int data)
{
    while (!(IFG1 & UTXIFG0))
        ;                /* USART0 TX buffer ready? */
    TXBUF0 = data;                             /* RXBUF0 to TXBUF0 */
}

#pragma vector = USART0RX_VECTOR
__interrupt void UART_0_RX_isr(void)
{
    while (!(IFG1 & UTXIFG0))
    {
        continue;
    }
}

void SendStringToBluetooth(char str[])
{
    int i;
    for (i = 0; i < strlen(str); i++)
    {
        SendUART(str[i]);
    }
}
```

Figure 40: UART for RN-42

These functions are used to communicate with the RN-42 bluetooth module through the UART pins. We had originally been using SendUART to send data one character at a time, but finally added SendStringToBluetooth to make our lives much easier.



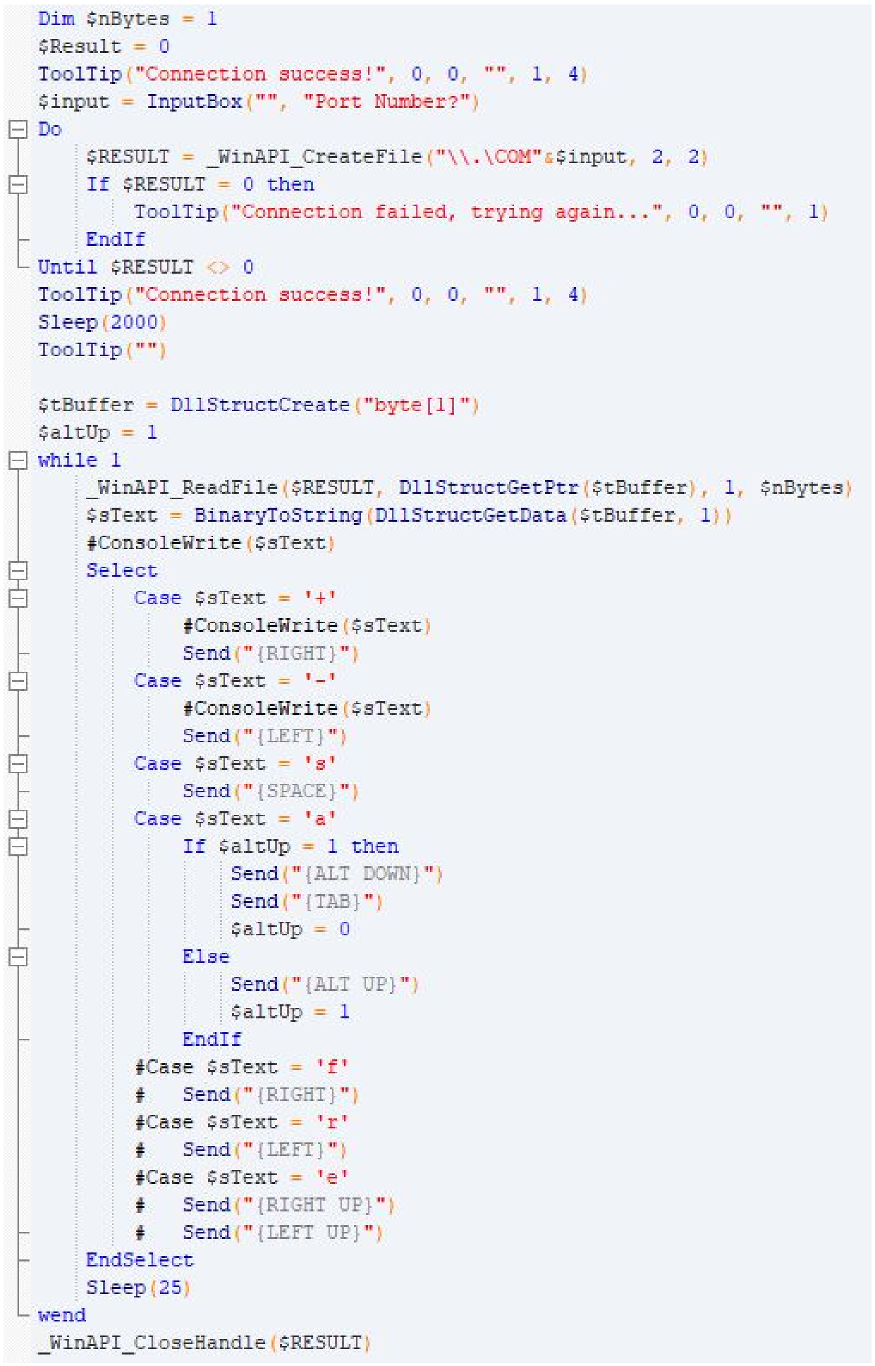
```
/* Detect left hand waving down and to the right */
if (gy - gy_bias < -force && az > 0)
{
    gesture_detected = gesture_time;
    vibe = vibe_time;
    SendStringToBluetooth("+\r\n");
    ClearMotionTracking();
    forwards = oneSecond;
}

/* Detect left hand waving up and to the left */
if (gy - gy_bias > force && az > 5000)
{
    gesture_detected = gesture_time;
    vibe = vibe_time;
    SendStringToBluetooth("-\r\n");
    ClearMotionTracking();
    backwards = oneSecond;
}

/* Detect left hand facing palm-up, then rising up */
if (gy - gy_bias < -force && az < -10000)
{
    gesture_detected = gesture_time;
    vibe = vibe_time;
    SendStringToBluetooth("a\r\n");
    ClearMotionTracking();
}
```

Figure 41: Interpreting Gestures

This is some of our gesture detection code. This also sends the command relating to the gesture out over bluetooth and sets the vibration timer to one half second.
‘force’ is used to detect if a sensor detected a value above a certain force threshold. We tested with various values, and a final implementation would certainly use different values for accelerometer, gyroscope, and along each axis - or a different approach entirely. However, this worked surprisingly well if only used to detect a handful of simple gestures.



```
Dim $nBytes = 1
$Result = 0
ToolTip("Connection success!", 0, 0, "", 1, 4)
$input = InputBox("", "Port Number?")
Do
    $RESULT = _WinAPI_CreateFile("\\.\COM"&$input, 2, 2)
    If $RESULT = 0 then
        ToolTip("Connection failed, trying again...", 0, 0, "", 1)
    EndIf
Until $RESULT <> 0
ToolTip("Connection success!", 0, 0, "", 1, 4)
Sleep(2000)
ToolTip("")

$tBuffer = DllStructCreate("byte[1]")
$altUp = 1
while 1
    _WinAPI_ReadFile($RESULT, DllStructGetPtr($tBuffer), 1, $nBytes)
    $sText = BinaryToString(DllStructGetData($tBuffer, 1))
    #ConsoleWrite($sText)
    Select
        Case $sText = '+'
            #ConsoleWrite($sText)
            Send("{RIGHT}")
        Case $sText = '-'
            #ConsoleWrite($sText)
            Send("{LEFT}")
        Case $sText = 's'
            Send("{SPACE}")
        Case $sText = 'a'
            If $altUp = 1 then
                Send("{ALT DOWN}")
                Send("{TAB}")
                $altUp = 0
            Else
                Send("{ALT UP}")
                $altUp = 1
            EndIf
        #Case $sText = 'f'
        #    Send("{RIGHT}")
        #Case $sText = 'r'
        #    Send("{LEFT}")
        #Case $sText = 'e'
        #    Send("{RIGHT UP}")
        #    Send("{LEFT UP}")
    EndSelect
    Sleep(25)
wend
_WinAPI_CloseHandle($RESULT)
```

Figure 42: AutoIt Driver

This is our simple driver code written in AutoIt - a microsoft windows automation language. This handles connecting to the watch over bluetooth, listening on the correct port, and reacting to input that comes in when a gesture is detected on the watch.

Watch and JTAG Interface Assembly

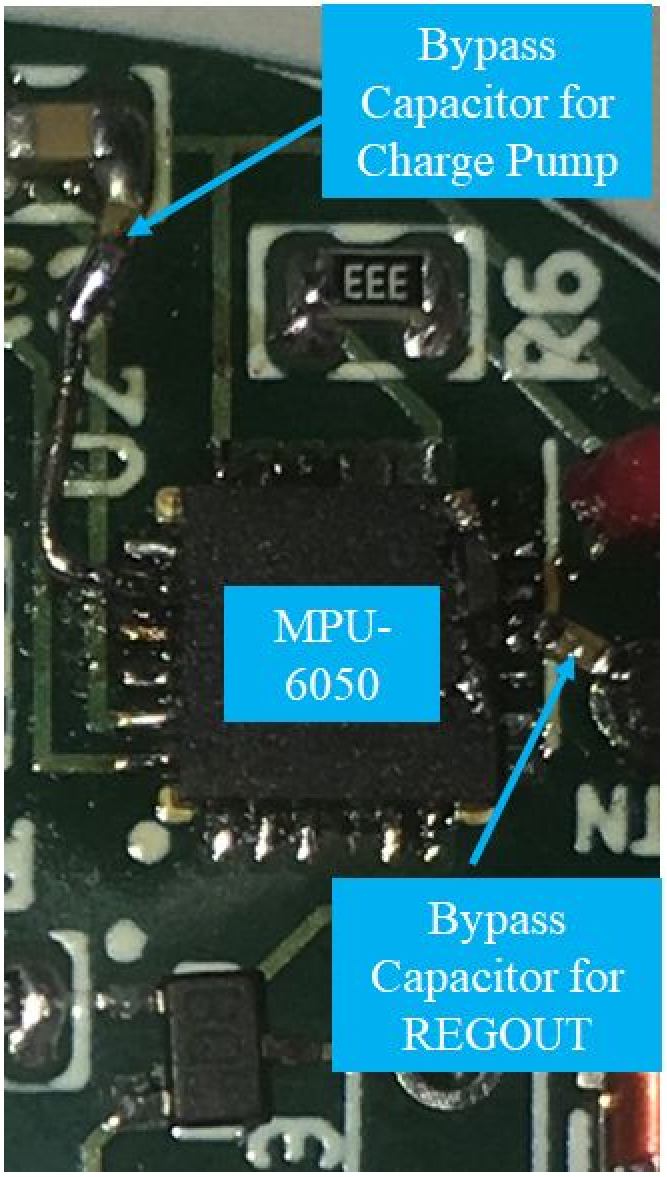


Figure 43: 3W Fixes

The above Figure 43 shows a closeup on the MPU-6050 of the final PCB. There were some unforeseen issues as the bypass capacitors for the regulator filter and charge pump were initially omitted. This was promptly fixed at 3W. Extra 0402 resistors were placed between existing components, with the regulator filter (REGOUT) being bypassed to the nearby ground through-hole of the pushbutton and the charge pump (CPOUT) being bypassed by wire to the neighboring SCL bypass capacitor.

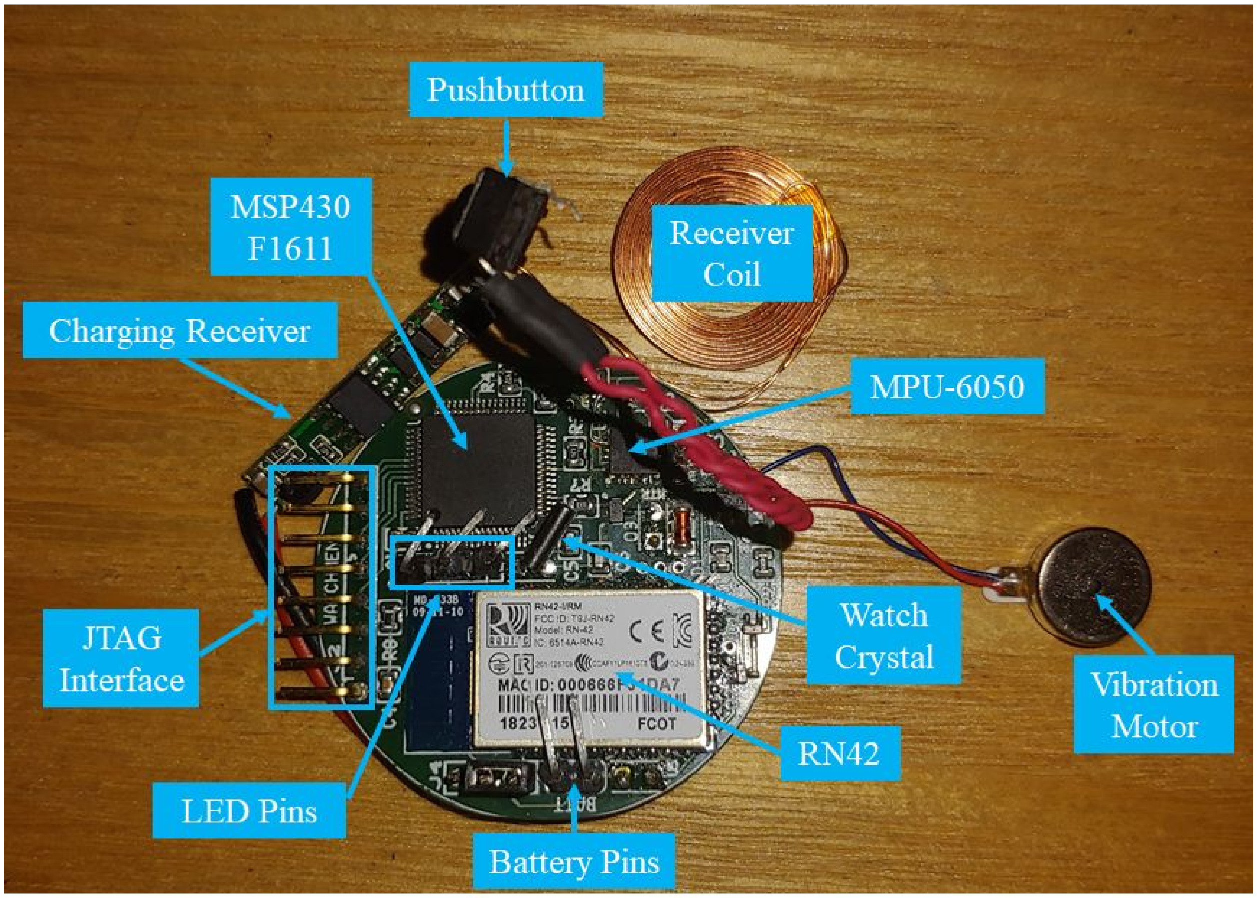


Figure 44: Final PCB without Chassis

The above Figure 44 shows the final mounted PCB with most of the throughhole components attached. Omitted are the LED display and battery. These are attached via the pins pictured and can easily be inserted and removed for assembly testing.

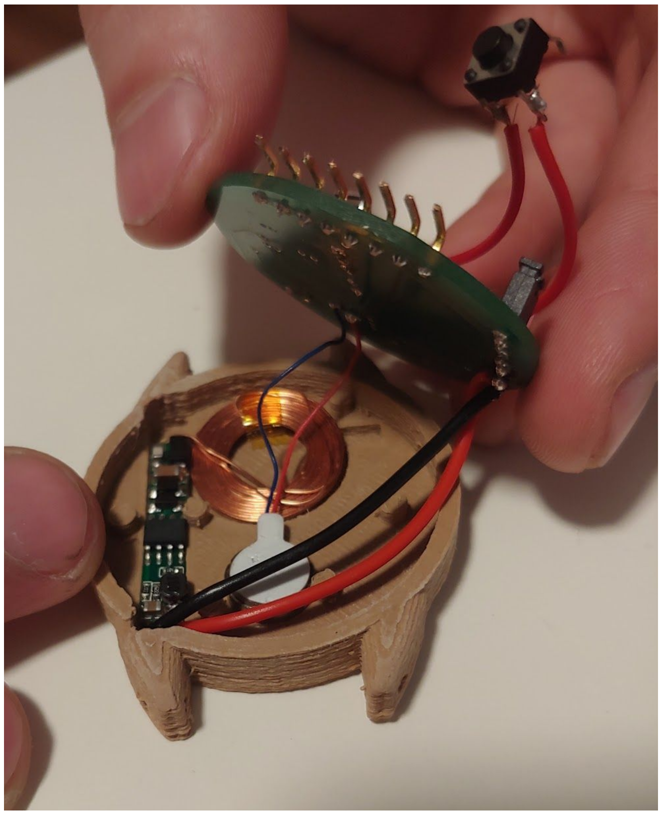


Figure 45: Assembling PCB and Bottom Chassis

Figure 45 shows the PCB and bottom chassis being assembled. The wireless charging receiver and vibration motor fit snugly and securely with the bottom cutouts.

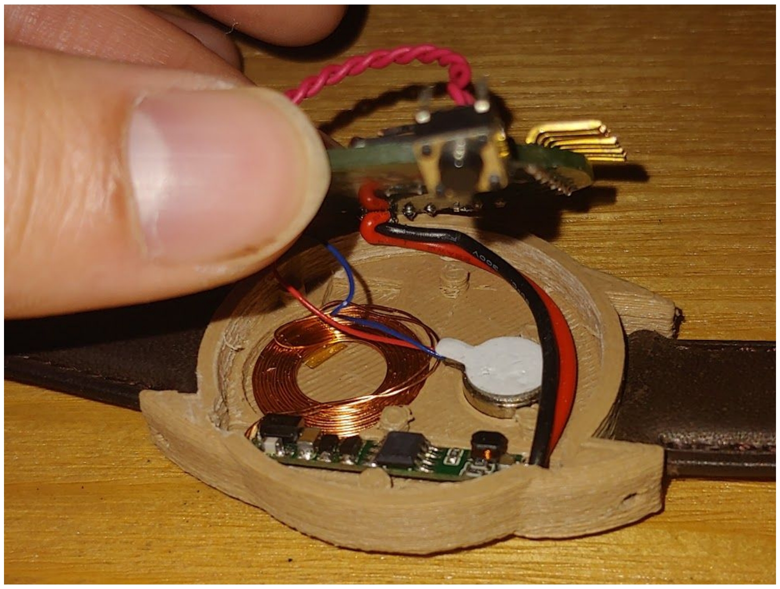


Figure 46: Assembling PCB and Bottom Chassis with Watch Band

Figure 46 shows the bottom chassis with the watch band attached.

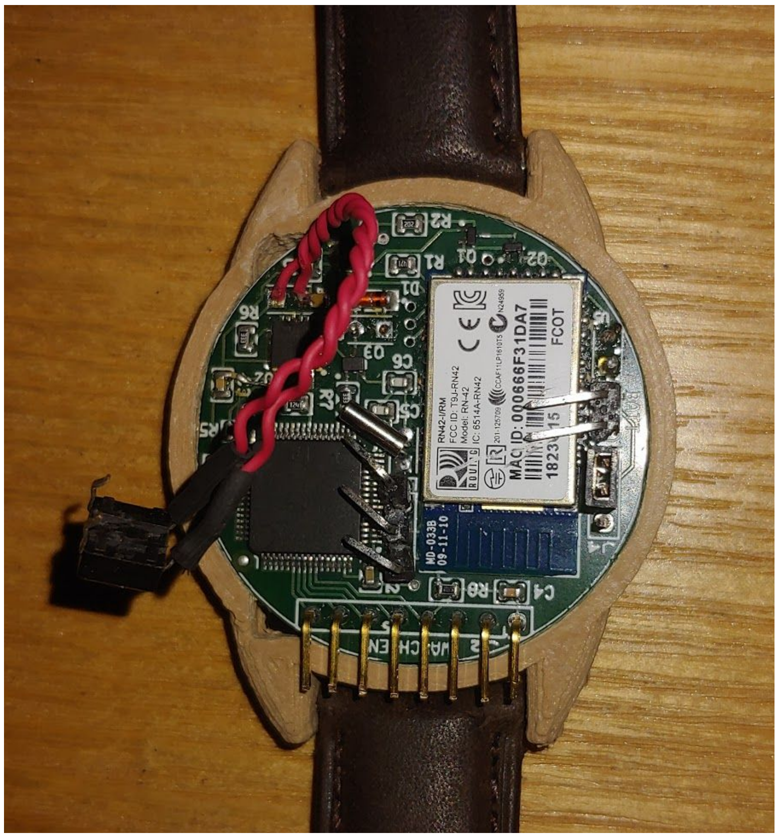


Figure 47: Final PCB in Bottom Chassis

Figure 47 shows the final PCB attached to the bottom chassis. The PCB sits flush with the surface of the chassis.

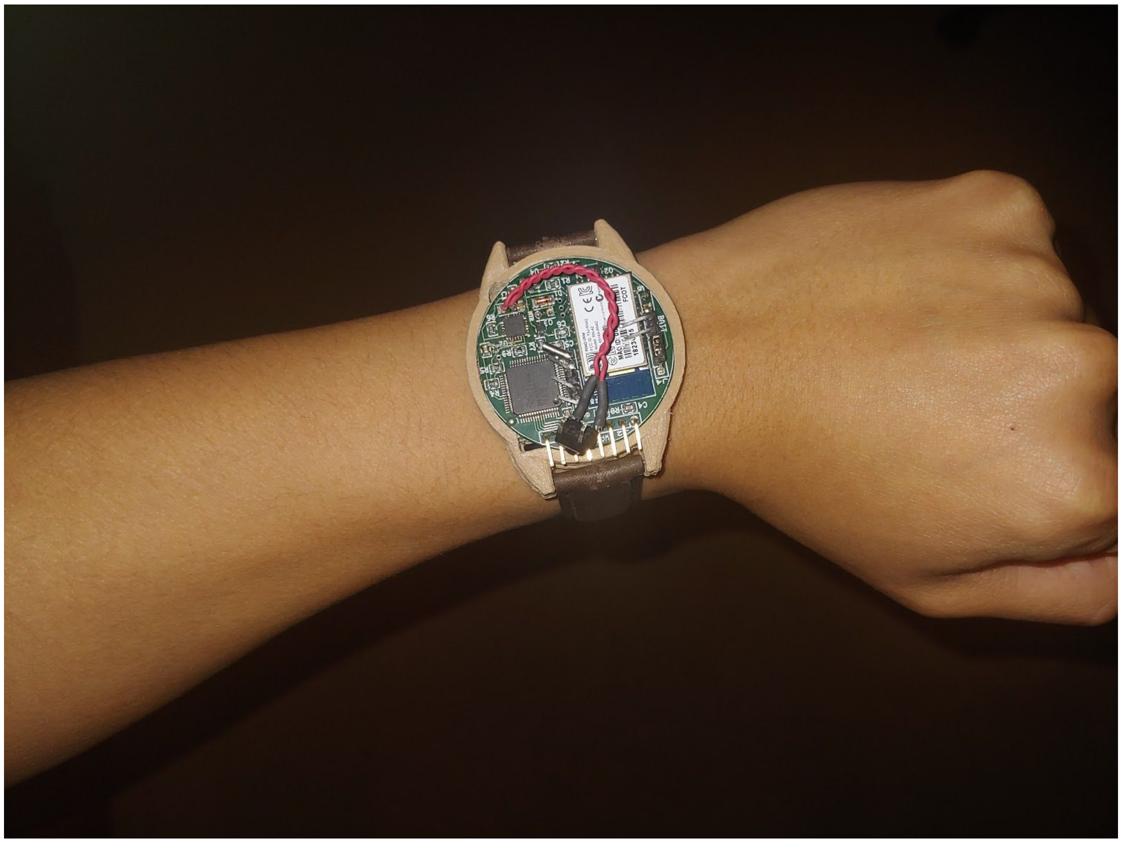


Figure 48: Bottom Chassis and Band Assembly Worn by User

Figure 48 above shows the watch being worn by a user without the top chassis attached.

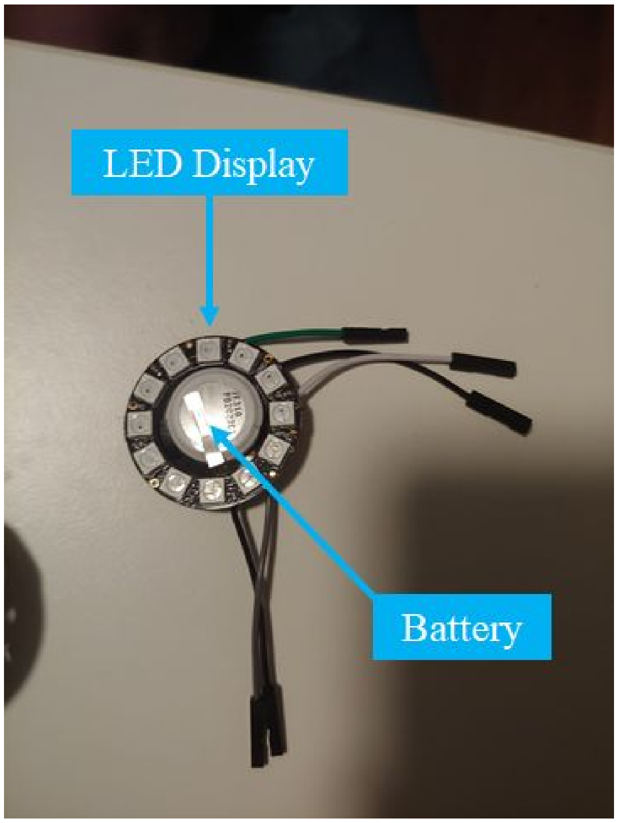


Figure 49: LED Ring and Battery

Figure 49 shows the LED ring with the battery taped to it.

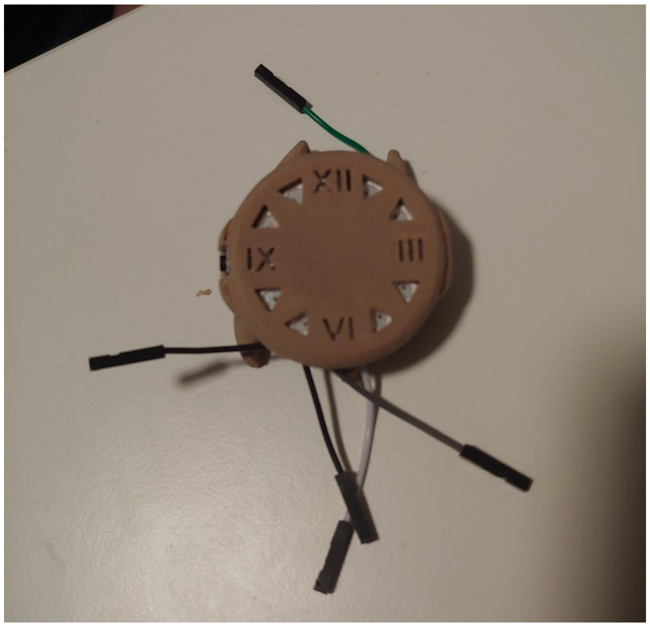


Figure 50: LED Ring and Battery with Top Chassis

Figure 50 shows the LED ring with the battery taped to it inside the top chassis.

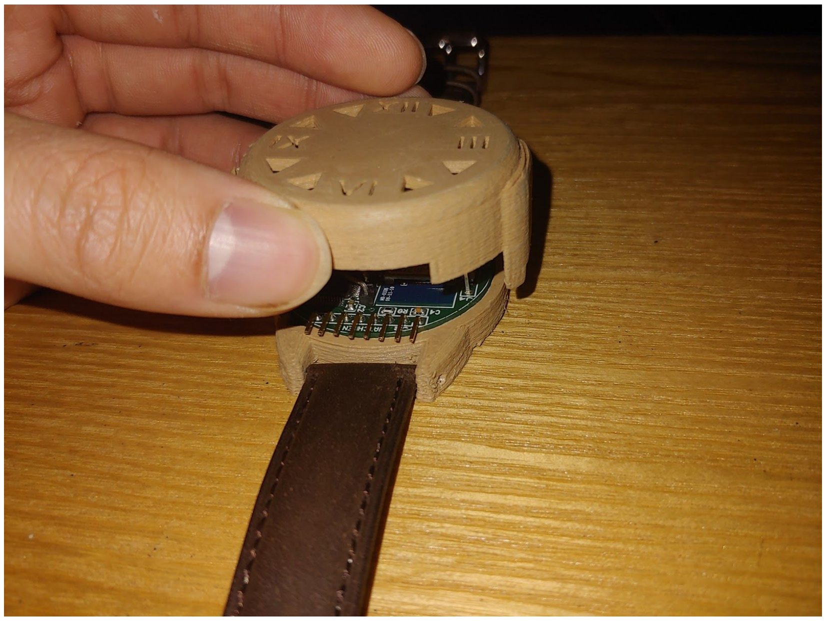


Figure 51: Enclosing Top/Bottom Chassis

Figure 51 shows the process used to enclose the watch.

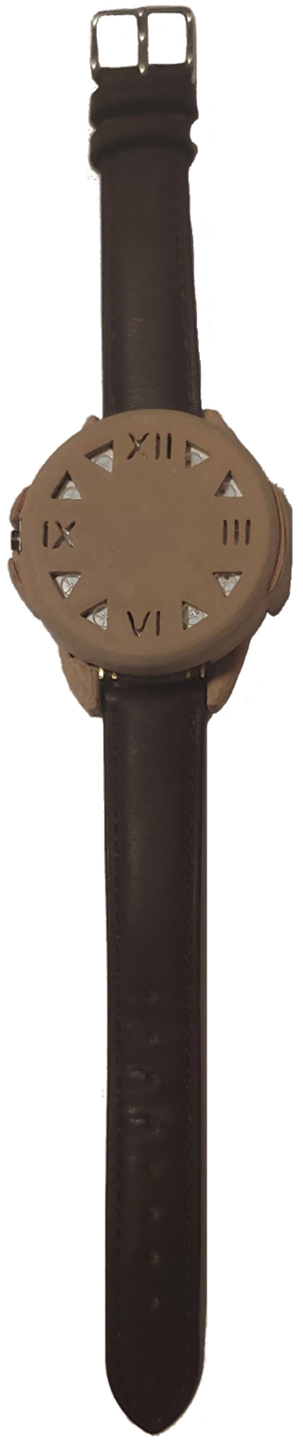


Figure 52: Final Assembled Watch

Figure 52 shows the final assembled watch.

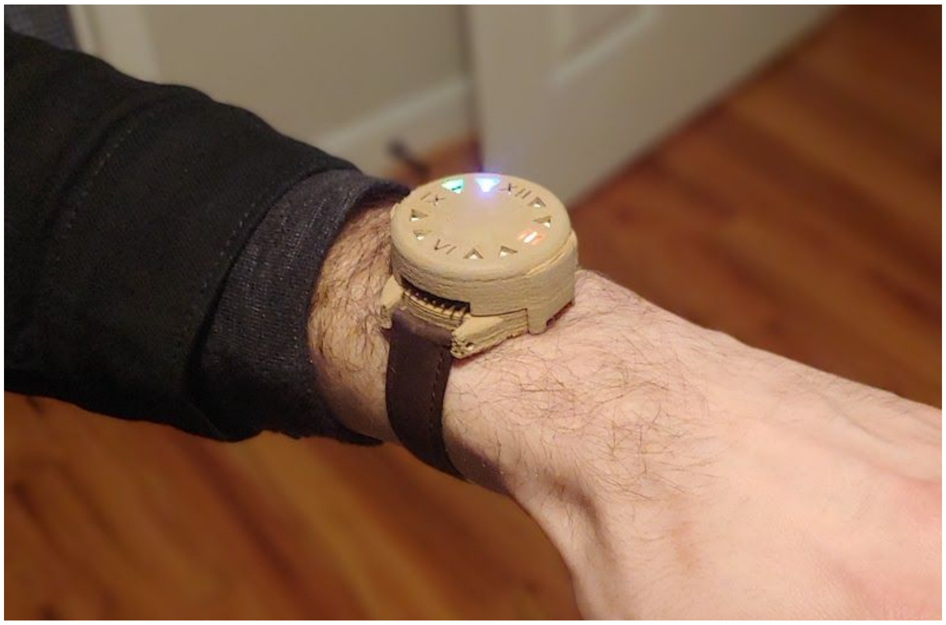


Figure 53: Final Assembled Watch Worn by User

Figure 53 shows the final assembled watch worn by a user.

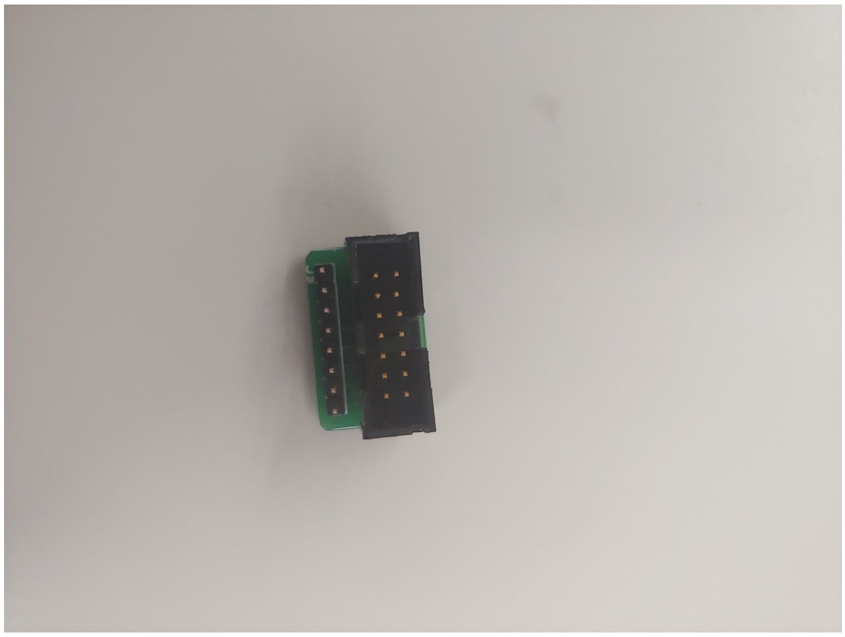


Figure 54: JTAG Helper Board

Figure 54 shows the JTAG helper board after it was cut out and the connection pins were soldered on.

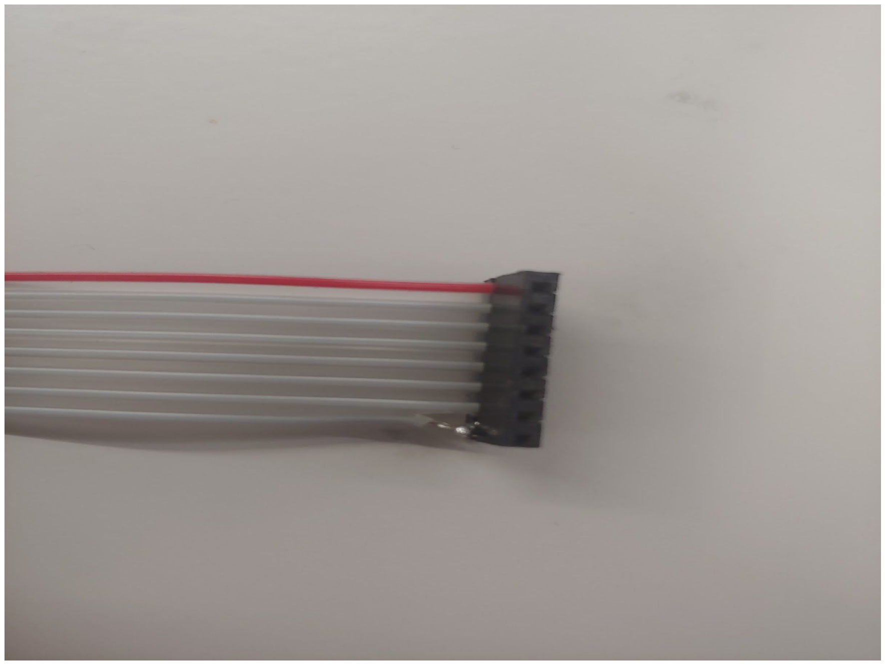


Figure 55: JTAG 8-Pin Ribbon Cable to Watch PCB

Figure 55 shows the JTAG 8-Pin Ribbon Cable that connects to the helper board. One of the pins was damaged in the process of debugging and had to be soldered back on to ensure a proper and reliable connection.

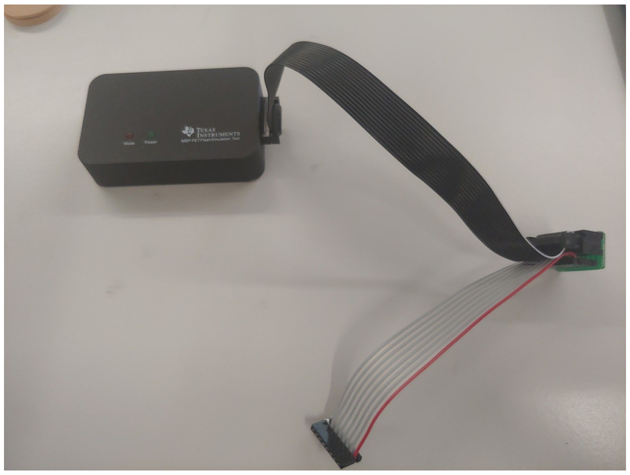


Figure 56: MSPFET with Helper JTAG Connection

Figure 56 shows the JTAG 8-Pin Ribbon Cable and MSPFET debugger connected to the JTAG helper board.

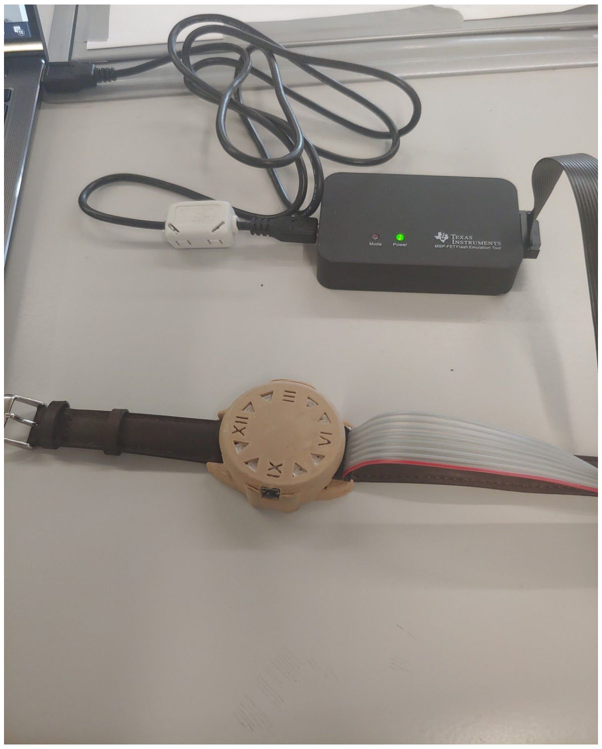


Figure 57: Watch Connected to Computer via MSPFET

Figure 57 shows the watch connected to the MSPFET debugger.

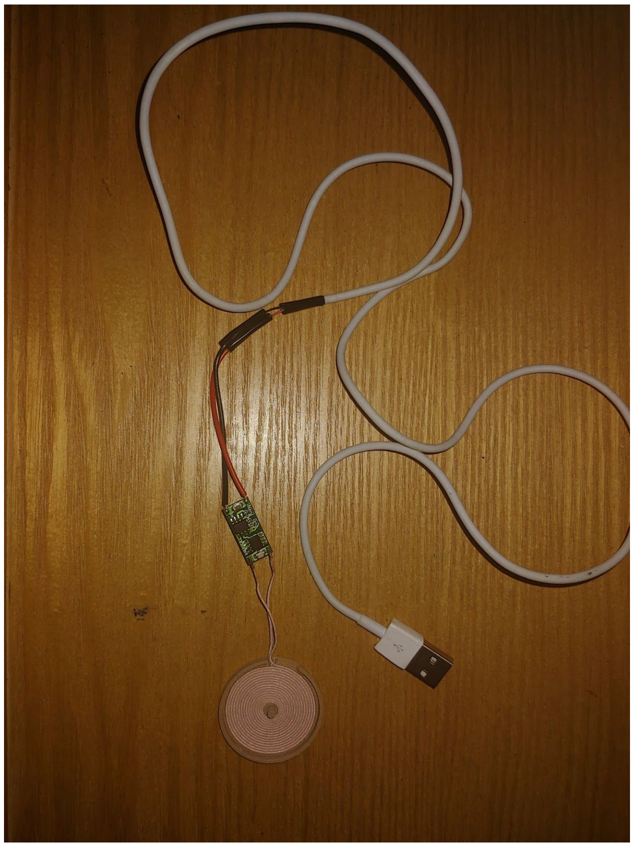


Figure 58: Inductive Charger Transceiver with Top Chassis Removed

Figure 58 shows the inductive charger transceiver with the top chassis removed. A USB cable was connected to the end of the transceiver to allow for ease of use.

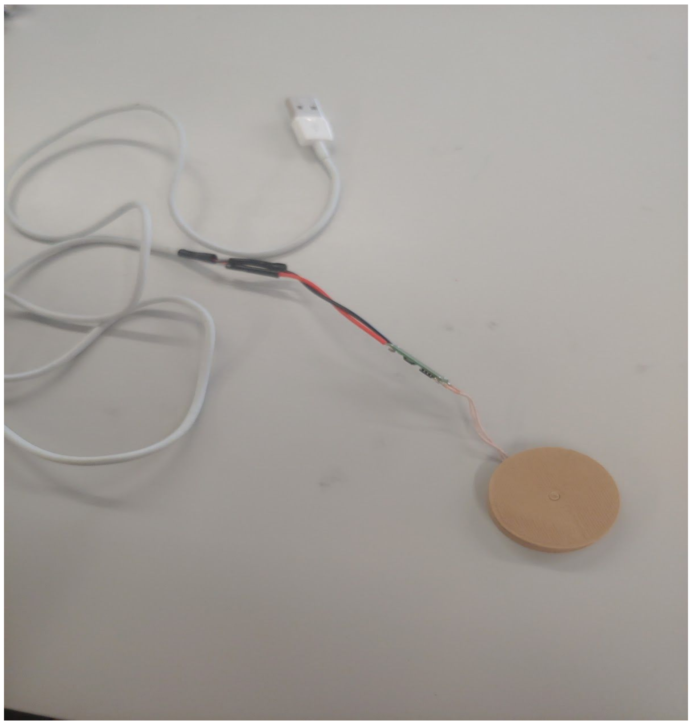


Figure 59: Complete Enclosed Inductive Charger Transceiver

Figure 59 shows the completely enclosed inductive charger transceiver.

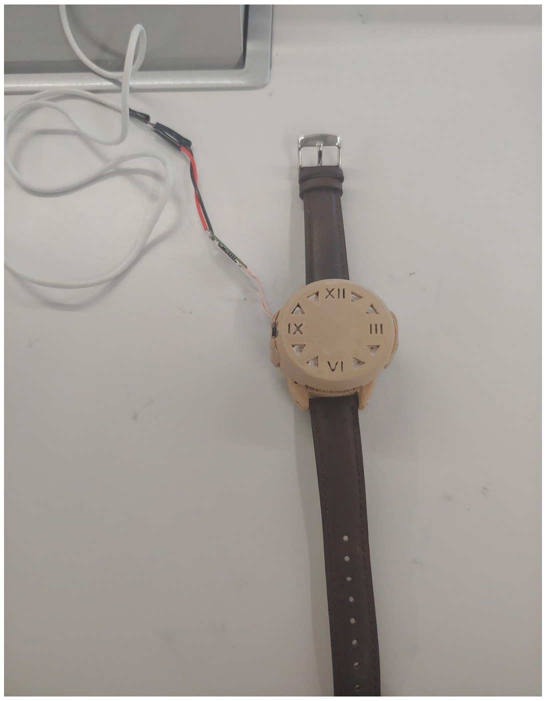


Figure 60: Watch Charging via Wireless Charging

Figure 60 shows the watch charging on the wireless charger.

Timekeeping Mode

Clicking the pushbutton will toggle between timekeeping mode, calibration mode, and gesture control mode. The timekeeping mode is pictured in Figure 61.

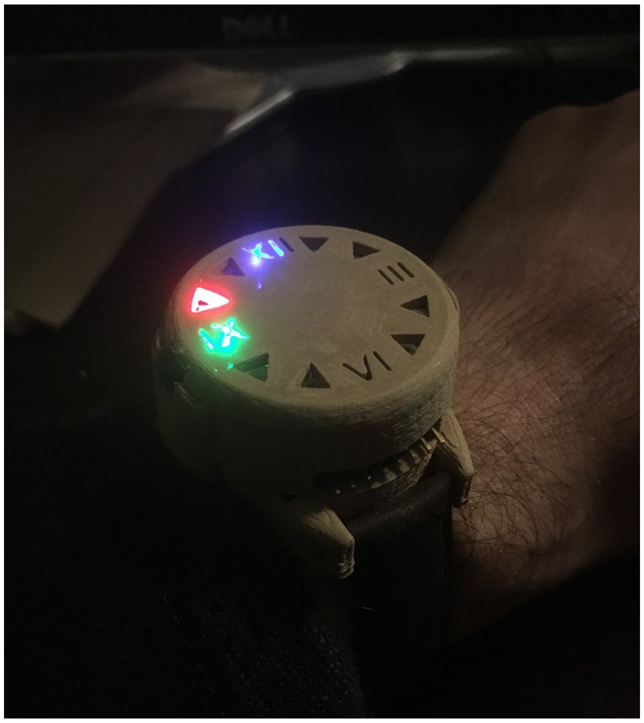


Figure 61: Timekeeping Mode

The red represents the hour hand, blue the minute, and green the second. Each green interval represents 25 seconds. This design choice was made because it allows for the most accurate timekeeping. 5 minutes divided by 12, for the number of LEDs, results in 25 seconds, hence the time can be told to an accuracy of within 25 seconds. See Figure 62 for clarification. The timekeeping mode uses a 32kHz watch crystal in order to ensure accurate time.

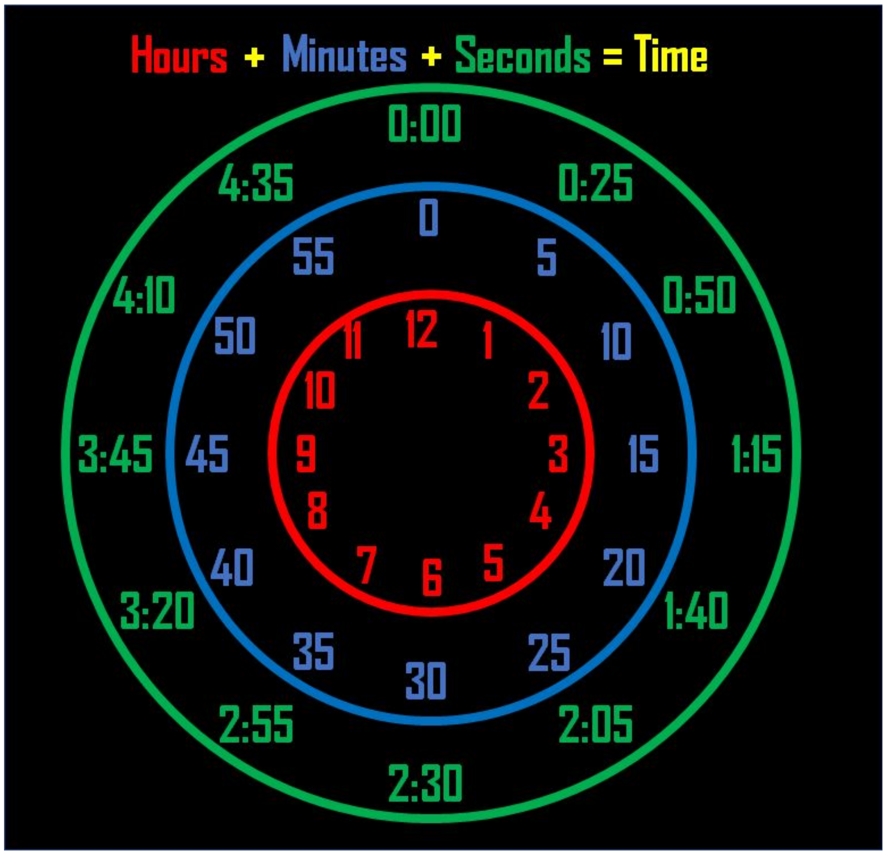


Figure 62: Timekeeping Guide Diagram

Calibration Mode

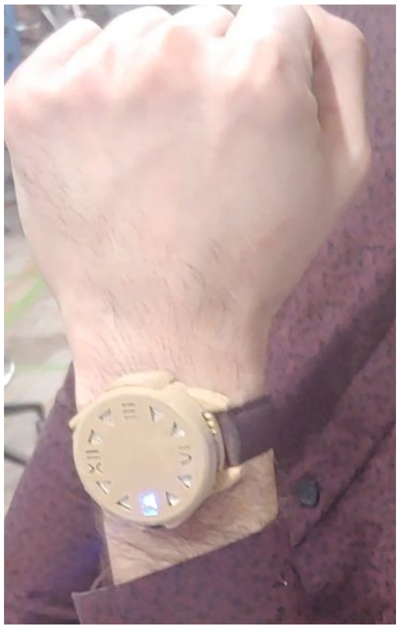


Figure 63: Calibration Mode

The above Figure 63 shows the calibration mode used for debugging the gyroscope/accelerometer. The 3, 6, 9, and 12 o'clock LEDs will turn blue when it is pointed directly toward the ground and the watch is held in an upright position with the ground.

Gesture Mode/ Coded Gestures

When the watch is roughly level with the watch face facing upwards, there are 4 green LEDs that show that the gyroscope/accelerometer is working as intended. See Figure 64. This is meant as a debugging failsafe in case the gyroscope/accelerometer were to malfunction. The gesture mode provides haptic feedback from the vibration motor when a gesture is successfully registered.

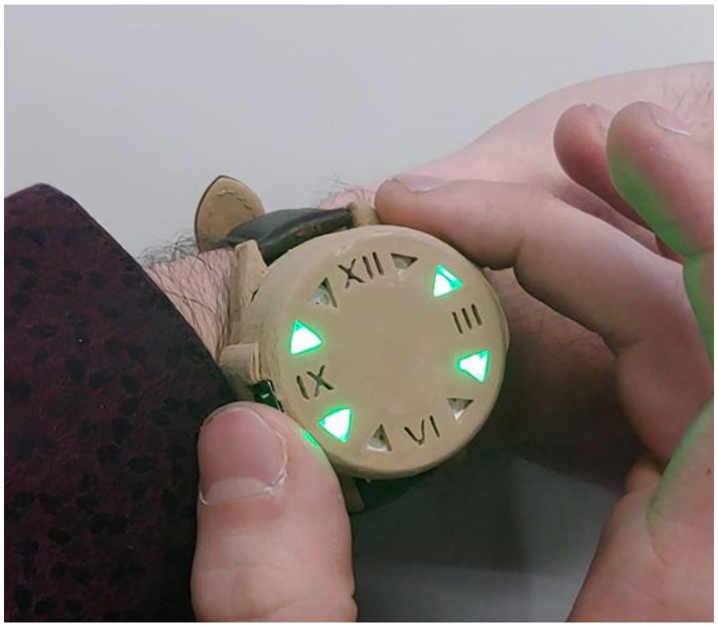
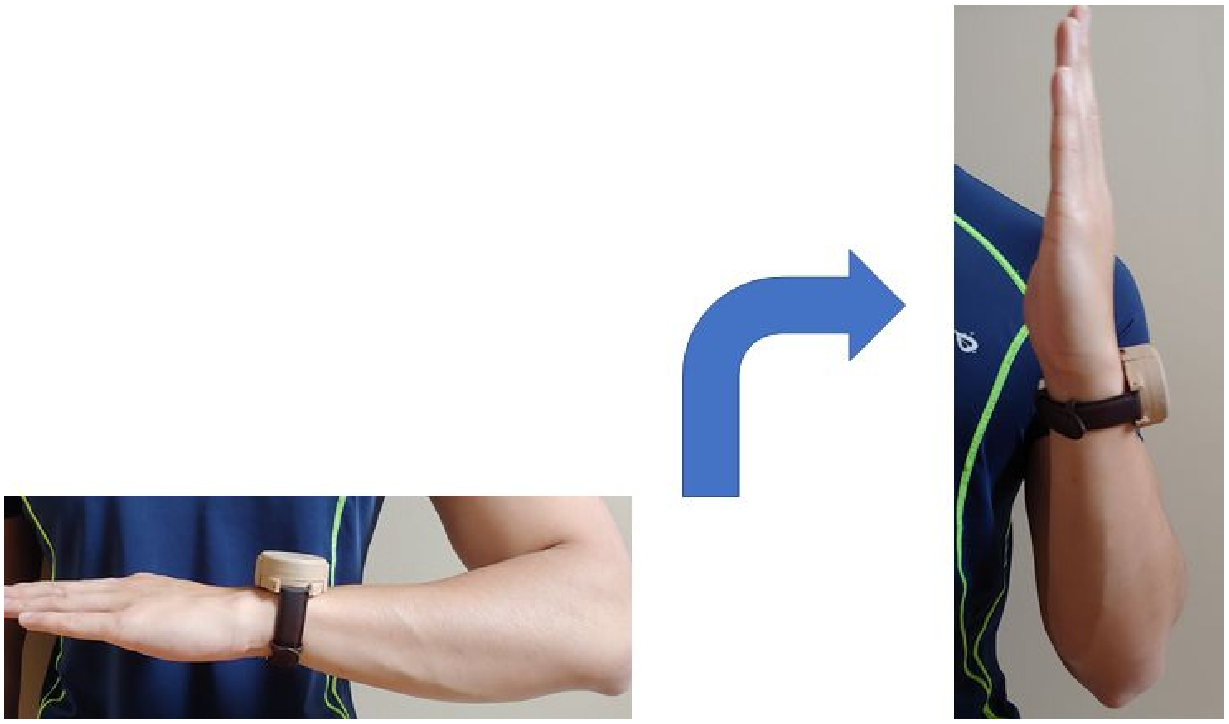
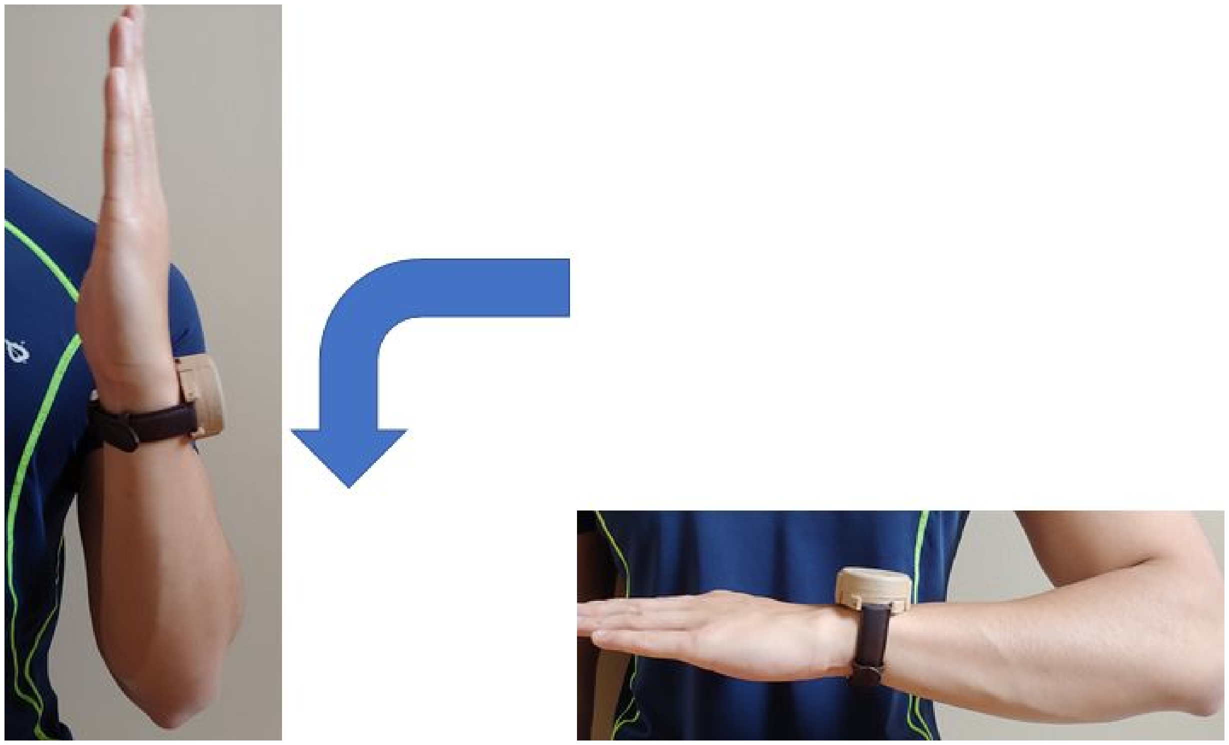


Figure 64: Gesture Mode Level Test


Backward/Left Gesture


Forward/Right Gesture

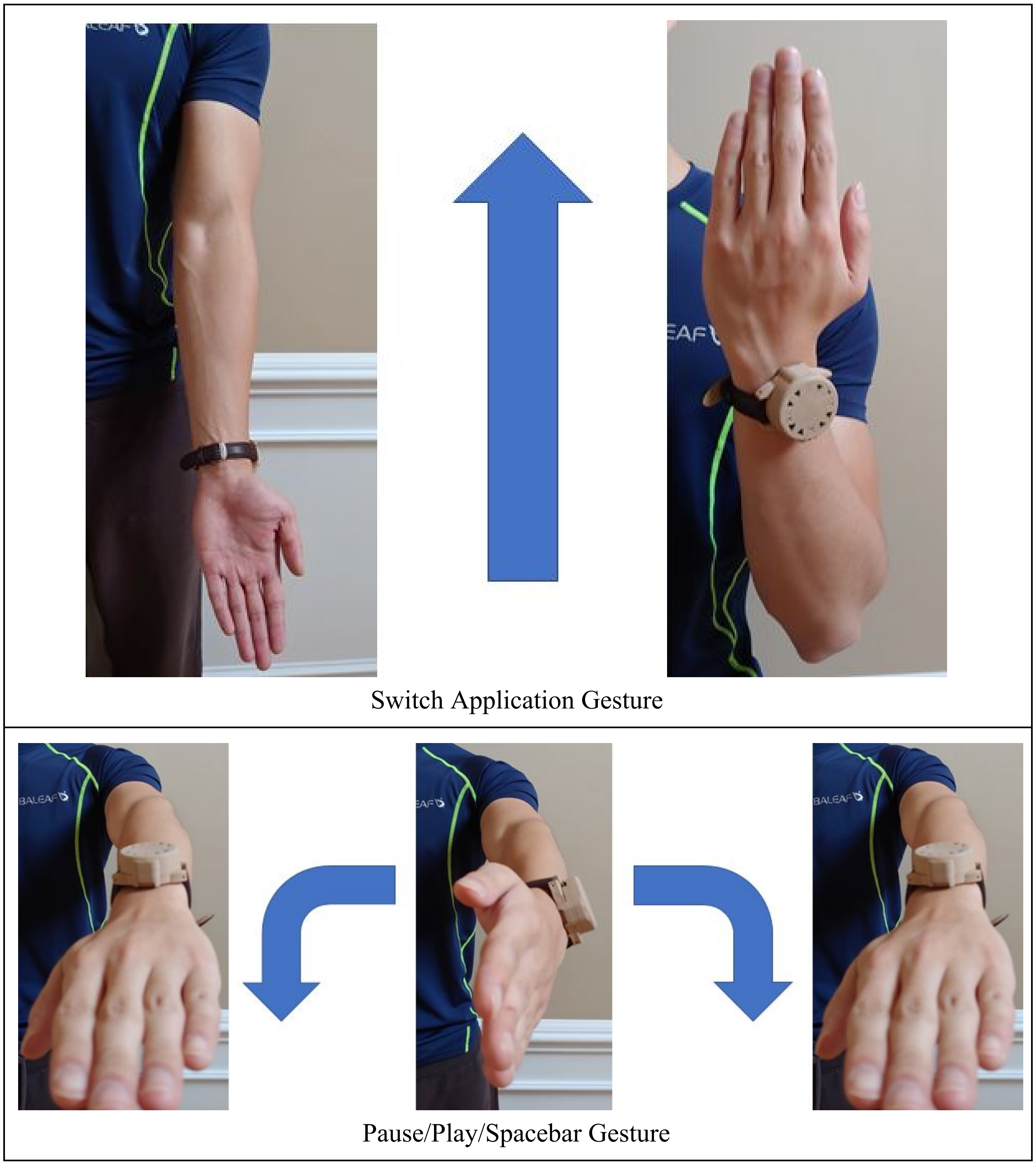
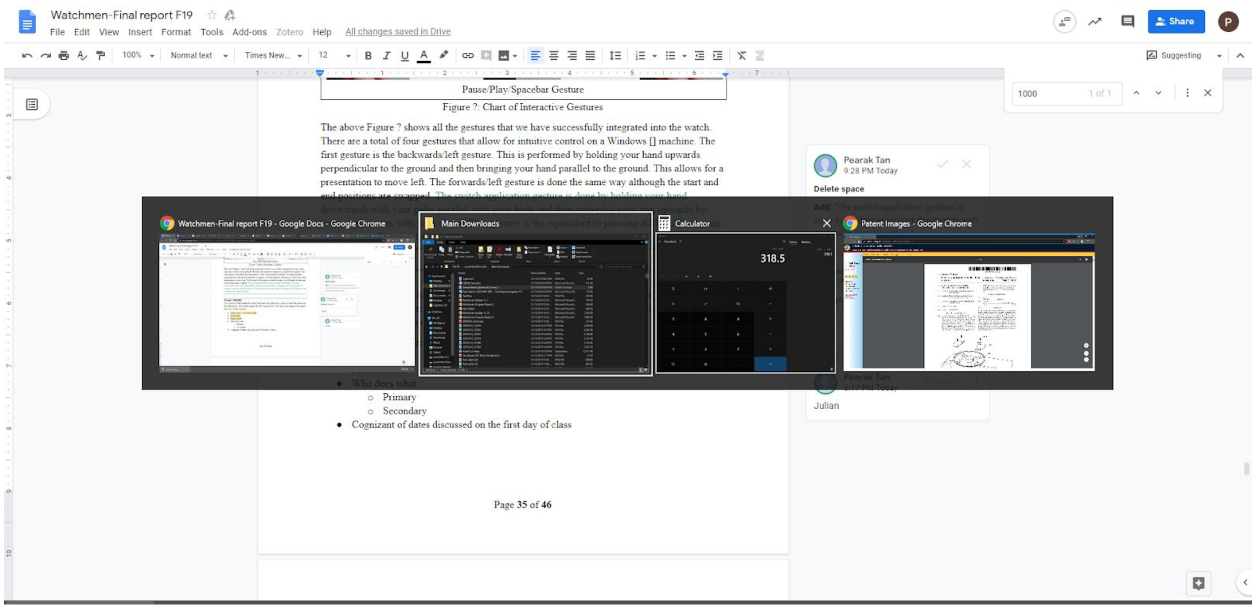


Figure 65: Chart of Interactive Gestures

The above Figure 65 shows all the gestures that we have successfully integrated into the watch. There are a total of four gestures that allow for intuitive control on a Windows machine. The first gesture is the backwards/left gesture. This is performed by holding your hand parallel to the ground and then bringing your hand upwards and perpendicular to the ground. This allows for a presentation to move left or a Youtube video to rewind five seconds.The forwards/left gesture is done the same way although the start and end positions are swapped. The switch application gesture is done by holding your hand downwards with your palm parallel with your body and then sweeping your arm upwards by rotating your arm with your elbow. This gesture is the equivalent to pressing ALT+TAB on your computer; see Figure 66 below.


Watchmen-Final report F19
File Edit View Insert Format Tools Add-ons Zotero Help
All changes saved in Drive
Share
Suggesting
Watchmen-Final report F19 - Google Docs - Google Chrome
Main Downloads
Calculator
318.5
Patent Images - Google Chrome

Pause/Play/Spacebar Gesture

Figure ?: Chart of Interactive Gestures

The above Figure ? shows all the gestures that we have successfully integrated into the watch. There are a total of four gestures that allow for intuitive control on a Windows [] machine. The first gesture is the backwards/left gesture. This is performed by holding your hand upwards perpendicular to the ground and then bringing your hand parallel to the ground. This allows for a presentation to move left. The forwards/left gesture is done the same way although the start and end positions are swapped. The switch application gesture is done by holding your hand

- Who does what
  - Primary
  - Secondary
- Cognizant of dates discussed on the first day of class

Page 35 of 46

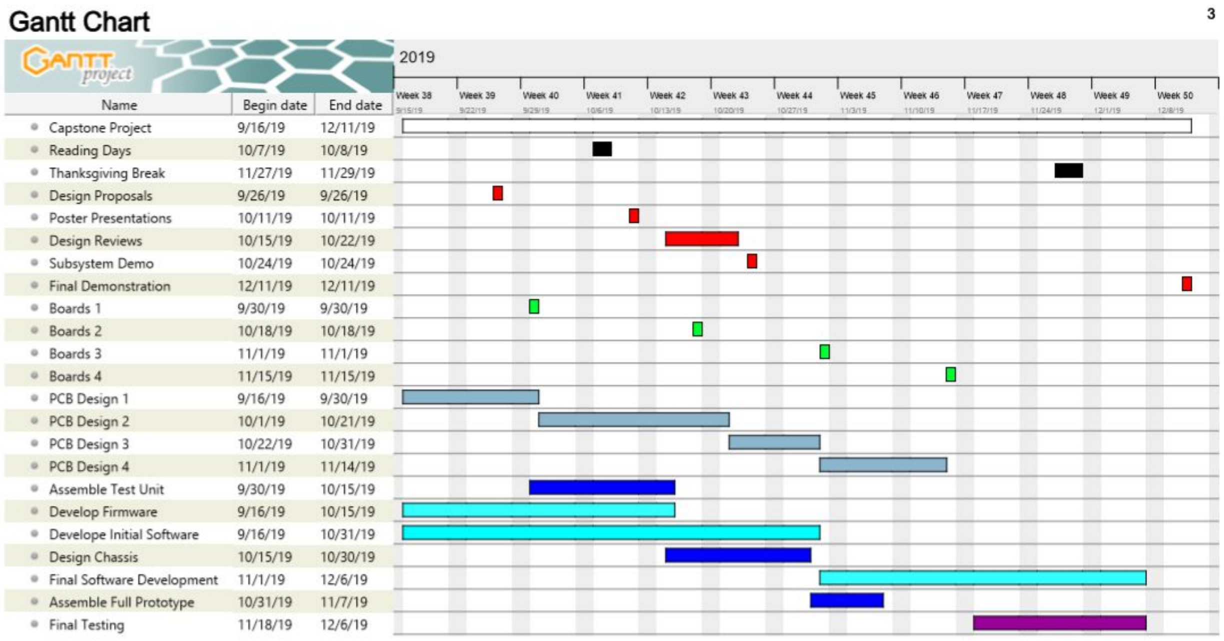
Pearak Tan
9:28 PM Today
Delete space

Julian

Figure 66: Switch Application Gesture Command

After this gesture is performed, using the aforementioned forward and backward gestures can be used to scroll through the applications. Performing the switch application gesture again will get you out of the Windows ALT+TAB mode and will open the application that is currently highlighted. The pause/play/spacebar gesture is used to pause and play media playback on the computer. This is intended to be mainly used for youtube playback and can be used to start and stop a video. It is performed by holding out your arm with the watch face situated upward and parallel to the ground and performing a 90 degree rotation to the outside of your body and another 90 degree rotation to return to the starting position.

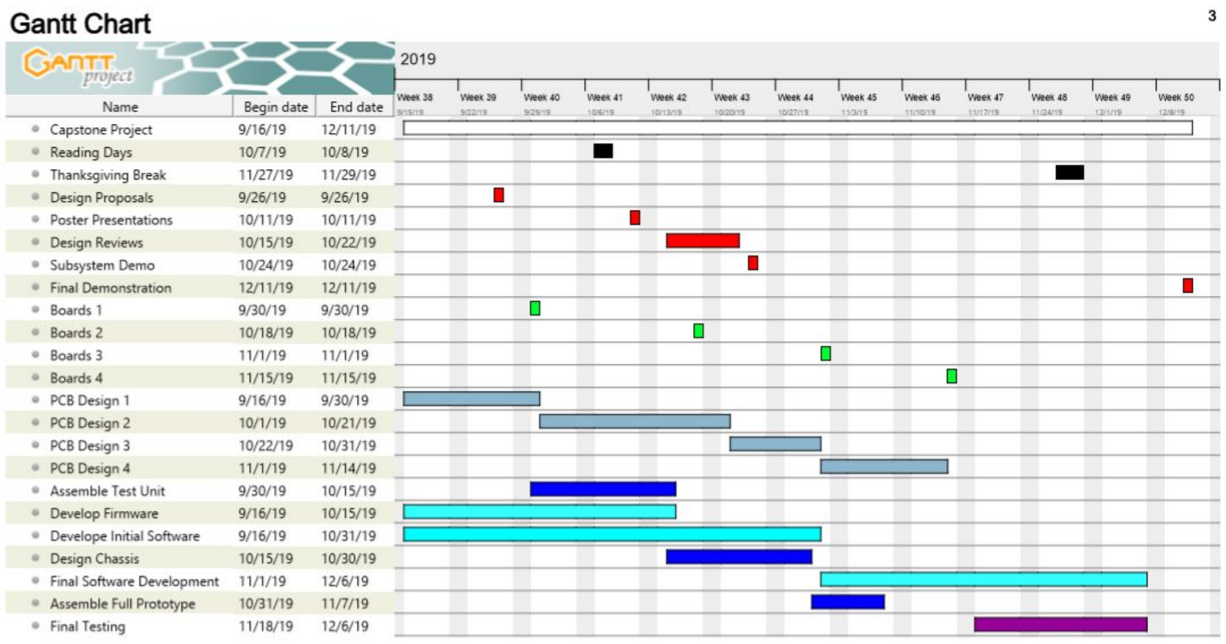
Project Timeline



Name	Begin date	End date
Capstone Project	9/16/19	12/11/19
Reading Days	10/7/19	10/8/19
Thanksgiving Break	11/27/19	11/29/19
Design Proposals	9/26/19	9/26/19
Poster Presentations	10/11/19	10/11/19
Design Reviews	10/15/19	10/22/19
Subsystem Demo	10/24/19	10/24/19
Final Demonstration	12/11/19	12/11/19
Boards 1	9/30/19	9/30/19
Boards 2	10/18/19	10/18/19
Boards 3	11/1/19	11/1/19
Boards 4	11/15/19	11/15/19
PCB Design 1	9/16/19	9/30/19
PCB Design 2	10/1/19	10/21/19
PCB Design 3	10/22/19	10/31/19
PCB Design 4	11/1/19	11/14/19
Assemble Test Unit	9/30/19	10/15/19
Develop Firmware	9/16/19	10/15/19
Develope Initial Software	9/16/19	10/31/19
Design Chassis	10/15/19	10/30/19
Final Software Development	11/1/19	12/6/19
Assemble Full Prototype	10/31/19	11/7/19
Final Testing	11/18/19	12/6/19

Figure 67: Original Gantt Chart

The chart pictured above in Figure 67 shows our original rough outline of our project schedule, deadlines and holidays. Much of this depended on when we acquired the parts we needed. Assembling a test unit with the MSP430, motion sensor, and battery wired together without a chassis was finished early. Assemblies with future PCB designs were dependent on when the PCB orders arrived. Coding software for final features depended on when the software/firmware handling baseline behavior was completely functional. We had multiple team members coding for various pieces of software simultaneously in order to allow for more efficient software development. Our first couple of PCB designs were missing vital components such as JTAG connectors and bypass capacitors, which led to many delays of the project as a whole. Our 3D modeling of the chassis was a trial and error process. Many chassis were printed and each iteration led to new shortcomings that had to be fixed. This further delayed the timeline.

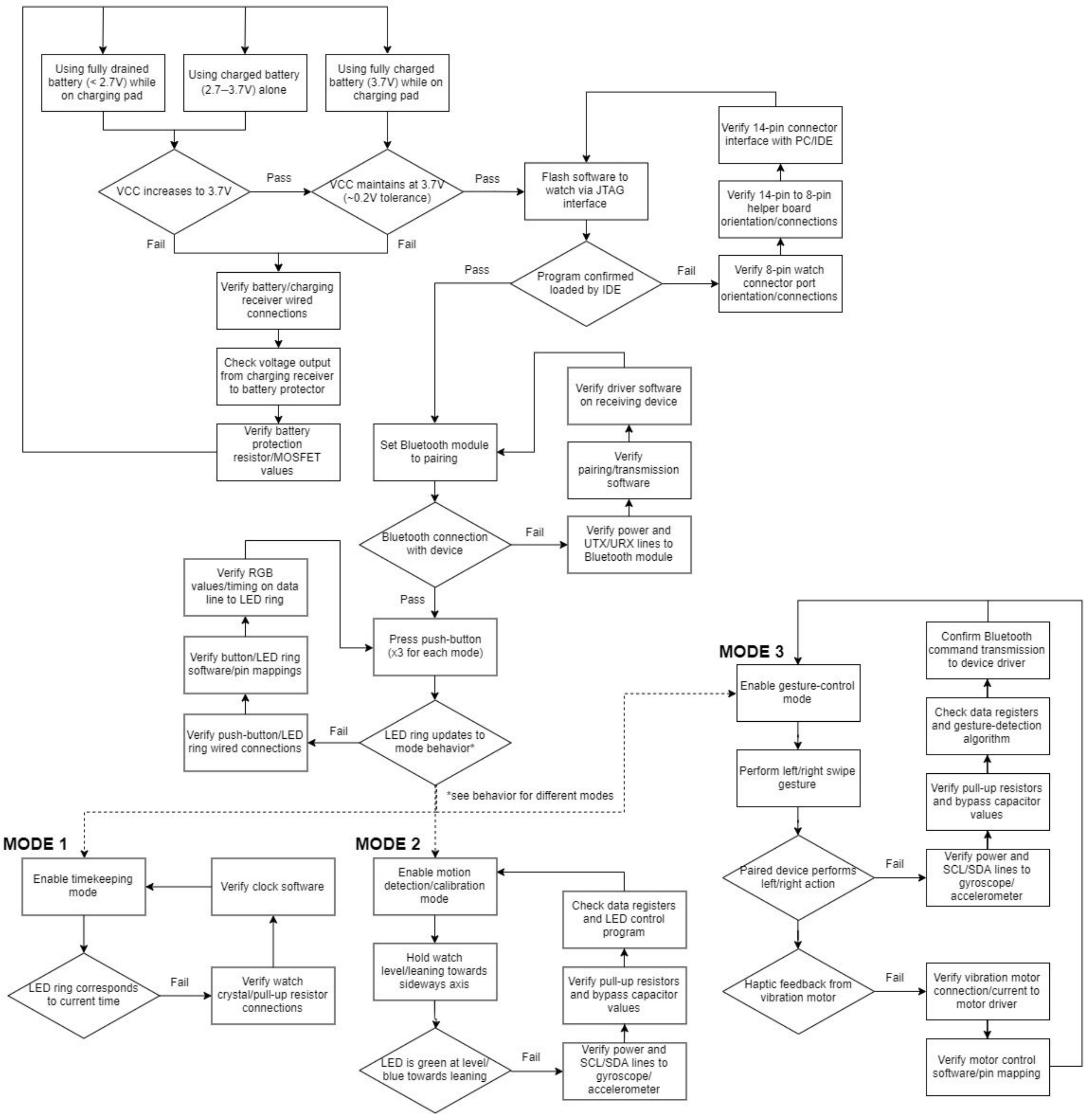


Name	Begin date	End date
Capstone Project	9/16/19	12/11/19
Reading Days	10/7/19	10/8/19
Thanksgiving Break	11/27/19	11/29/19
Design Proposals	9/26/19	9/26/19
Poster Presentations	10/11/19	10/11/19
Design Reviews	10/15/19	10/22/19
Subsystem Demo	10/24/19	10/24/19
Final Demonstration	12/11/19	12/11/19
Boards 1	9/30/19	9/30/19
Boards 2	10/18/19	10/18/19
Boards 3	11/1/19	11/1/19
Boards 4	11/15/19	11/15/19
PCB Design 1	9/16/19	9/30/19
PCB Design 2	10/1/19	10/21/19
PCB Design 3	10/22/19	10/31/19
PCB Design 4	11/1/19	11/14/19
Assemble Test Unit	9/30/19	10/15/19
Develop Firmware	9/16/19	10/15/19
Develope Initial Software	9/16/19	10/31/19
Design Chassis	10/15/19	10/30/19
Final Software Development	11/1/19	12/6/19
Assemble Full Prototype	10/31/19	11/7/19
Final Testing	11/18/19	12/6/19

Figure 68: Finalized Gantt Chart

Figure 68 above shows the finalized Gantt chart. Much of it is the same as the original, although more time had to be allotted for printing the watch chassis and mounting. Some things had to be delayed simply because the parts did not arrive in a timely manner.

Test Plan



Using fully drained battery (< 2.7V) while on charging pad
Using charged battery (2.7–3.7V) alone
Using fully charged battery (3.7V) while on charging pad
VCC increases to 3.7V
Pass
VCC maintains at 3.7V (~0.2V tolerance)
Pass
Fail
Fail
Flash software to watch via JTAG interface
Verify 14-pin connector interface with PC/IDE
Verify 14-pin to 8-pin helper board orientation/connections
Verify 8-pin watch connector port orientation/connections
Program confirmed loaded by IDE
Pass
Fail
Verify battery/charging receiver wired connections
Check voltage output from charging receiver to battery protector
Verify battery protection resistor/MOSFET values
Set Bluetooth module to pairing
Verify driver software on receiving device
Verify pairing/transmission software
Verify power and UTX/URX lines to Bluetooth module
Bluetooth connection with device
Fail
Pass
Verify RGB values/timing on data line to LED ring
Verify button/LED ring software/pin mappings
Verify push-button/LED ring wired connections
Press push-button (x3 for each mode)
LED ring updates to mode behavior*
Fail
*see behavior for different modes
MODE 1
Enable timekeeping mode
Verify clock software
LED ring corresponds to current time
Fail
Verify watch crystal/pull-up resistor connections
MODE 2
Enable motion detection/calibration mode
Hold watch level/leaning towards sideways axis
LED is green at level/ blue towards leaning
Fail
Check data registers and LED control program
Verify pull-up resistors and bypass capacitor values
Verify power and SCL/SDA lines to gyroscope/ accelerometer
MODE 3
Enable gesture-control mode
Perform left/right swipe gesture
Paired device performs left/right action
Fail
Haptic feedback from vibration motor
Fail
Confirm Bluetooth command transmission to device driver
Check data registers and gesture-detection algorithm
Verify pull-up resistors and bypass capacitor values
Verify power and SCL/SDA lines to gyroscope/ accelerometer
Verify vibration motor connection/current to motor driver
Verify motor control software/pin mapping

Figure 69: Watch Test Plan

Figure 69 illustrates the test plan for verifying all of the watch's subsystems and software modes. It starts by first confirming basic functionality of the power module, JTAG interface, and Bluetooth connectivity. Afterwards it evaluates the behavior of the 3 different usage modes in context of visual, haptic, and Bluetooth paired feedback.

Power Module Testing

Verifying the power module consists of measuring the VCC supply voltage to the system in various battery/charging conditions.

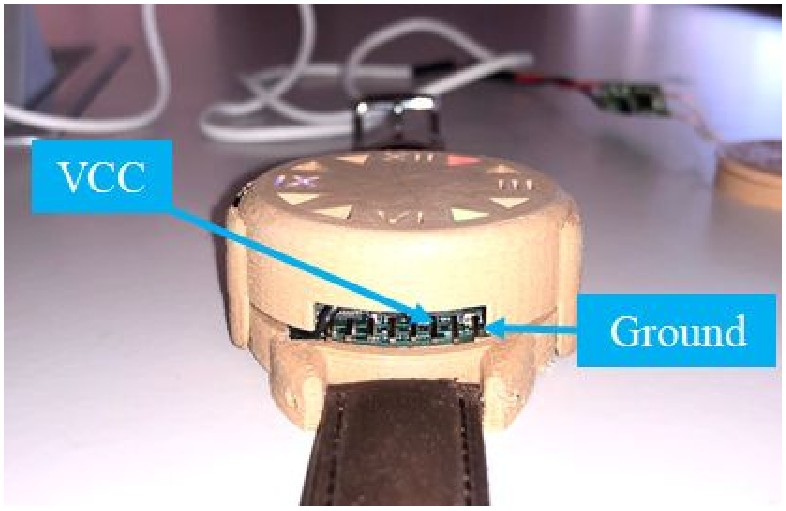


Figure 70: JTAG Port Pins for Power Module Testing

To easily measure the supply voltage to the system, we applied a multimeter to the JTAG interface pins corresponding to VCC and ground, as shown in Figure 70.

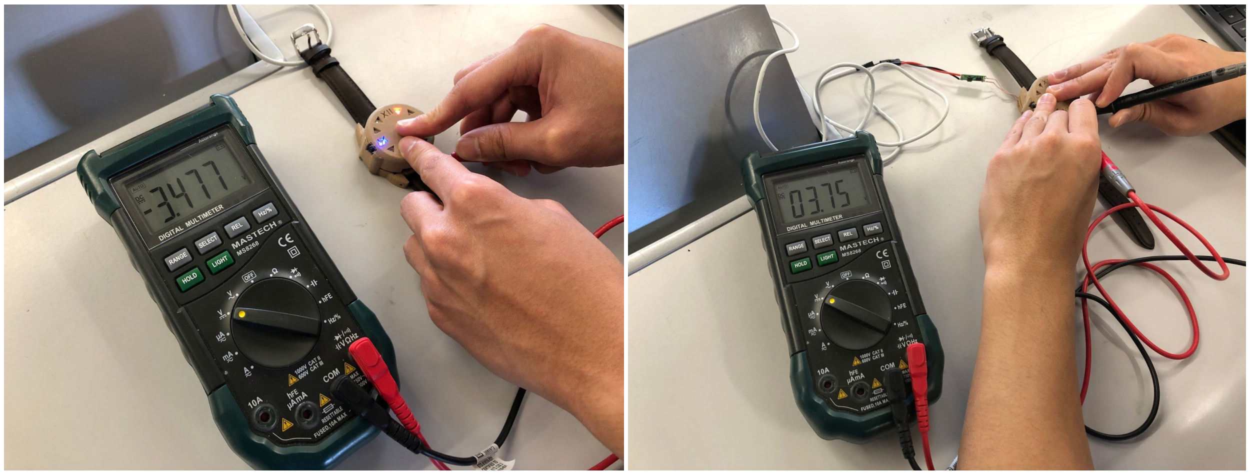


Figure 71: Battery Charging Verification

Figure 71 (left) shows the 3.7V battery having been slightly drained to ~3.5V and subsequently recharged to ~3.7V (right) after being placed on the wireless charging transceiver for some time. (Note the charging transceiver's wires leading to the watch in the right image)

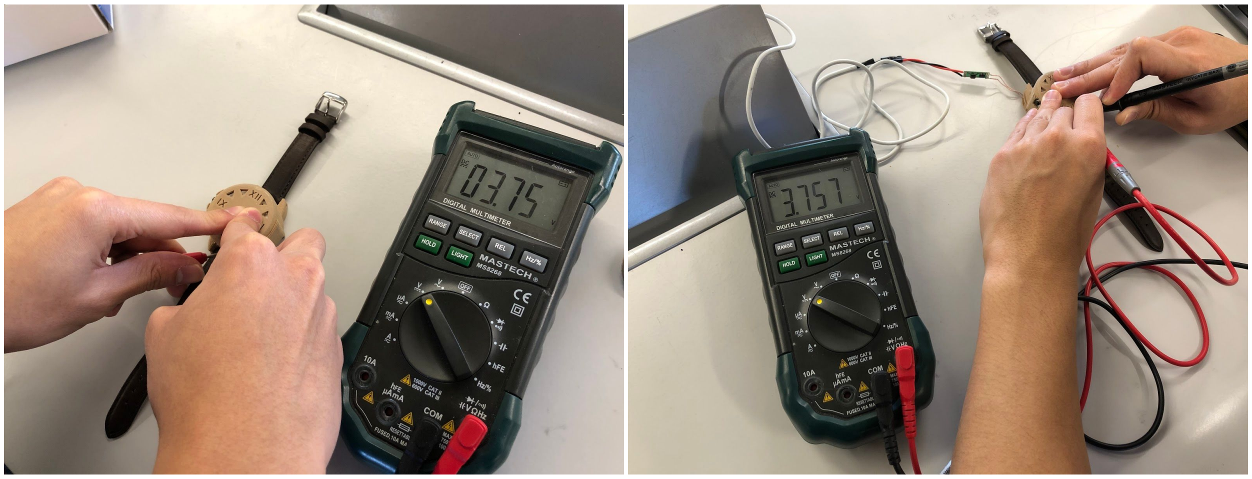


Figure 72: Battery Protection Verification

Figure 72 (left) shows the battery starting at fully charged capacity. Figure 72 (right) shows the battery's voltage plateauing around 3.7V even while placed on the wireless charger.

After confirming both charging and battery protection functionality, we could move on to testing other components dependent on the power supply.

JTAG Testing

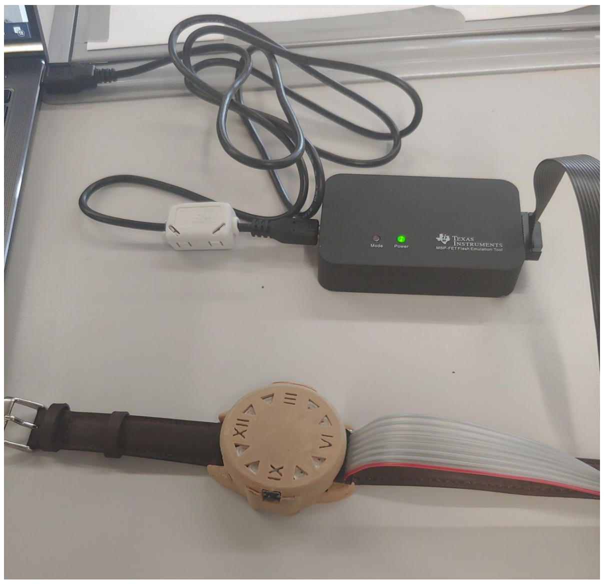
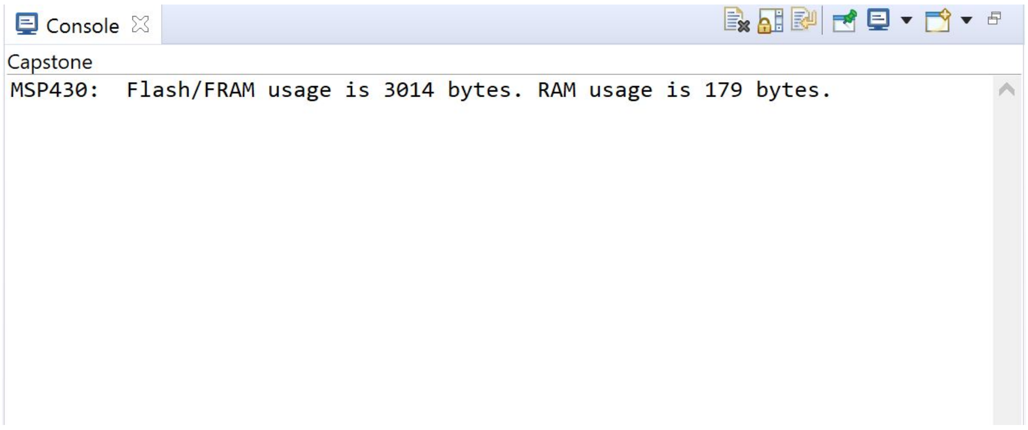


Figure 73: Connecting Watch to PC via JTAG Interface

Figure 73 shows the watch's JTAG port connecting to the development debugging kit through the specialized helper board interface.

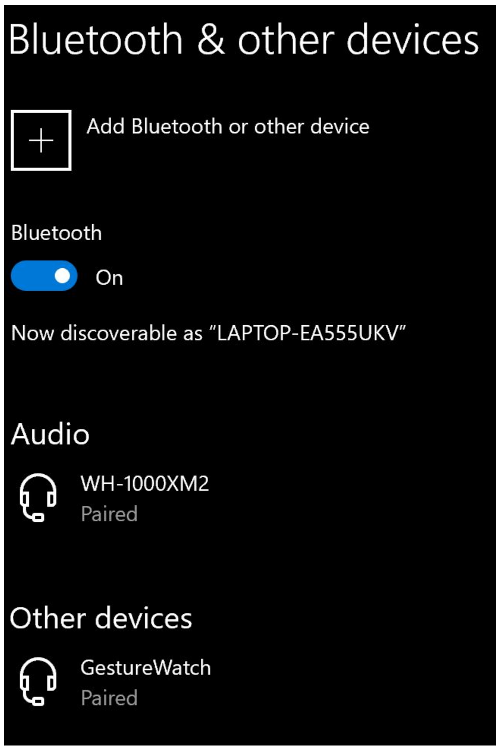


```
Capstone
MSP430:  Flash/FRAM usage is 3014 bytes. RAM usage is 179 bytes.
```

Figure 74: Flashing Firmware to Watch via Code Composer

Figure 74 shows the confirmation message in Code Composer that the software has uploaded to the MSP430 successfully.

Bluetooth Testing


Bluetooth & other devices

Bluetooth: On

Now discoverable as "LAPTOP-EA555UKV"

Audio

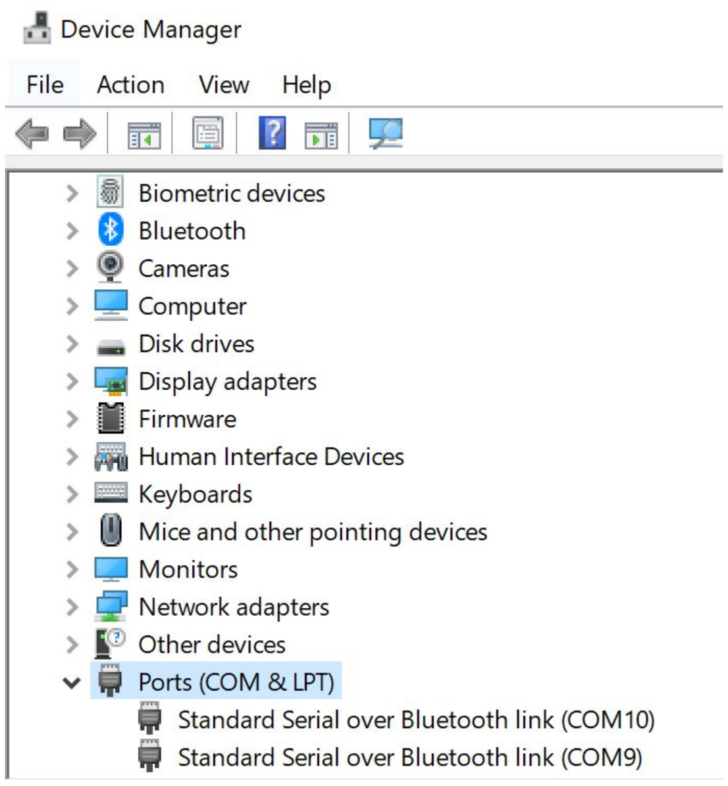
WH-1000XM2 — Paired

Other devices

GestureWatch — Paired

Figure 75: Bluetooth Pairing with Windows PC

Upon powering on and flashing the software, the watch began advertising on all nearby Bluetooth-enabled devices. Figure 75 shows a Windows 10 PC successfully paired with the watch, designated as "GestureWatch".



Biometric devices
Bluetooth
Cameras
Computer
Disk drives
Display adapters
Firmware
Human Interface Devices
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Other devices
Ports (COM & LPT)
Standard Serial over Bluetooth link (COM10)
Standard Serial over Bluetooth link (COM9)

Figure 76: Verifying COM Port Connection

Figure 76 shows the confirmation of an opened COM port connection with the watch for Bluetooth communication.

Mode Testing

The single function of the pushbutton was verified by successfully toggling through each software mode. Each of the 3 modes were then independently verified by observing the expected operation of motion detecting and feedback systems, thus also verifying their constituent subsystems and software.

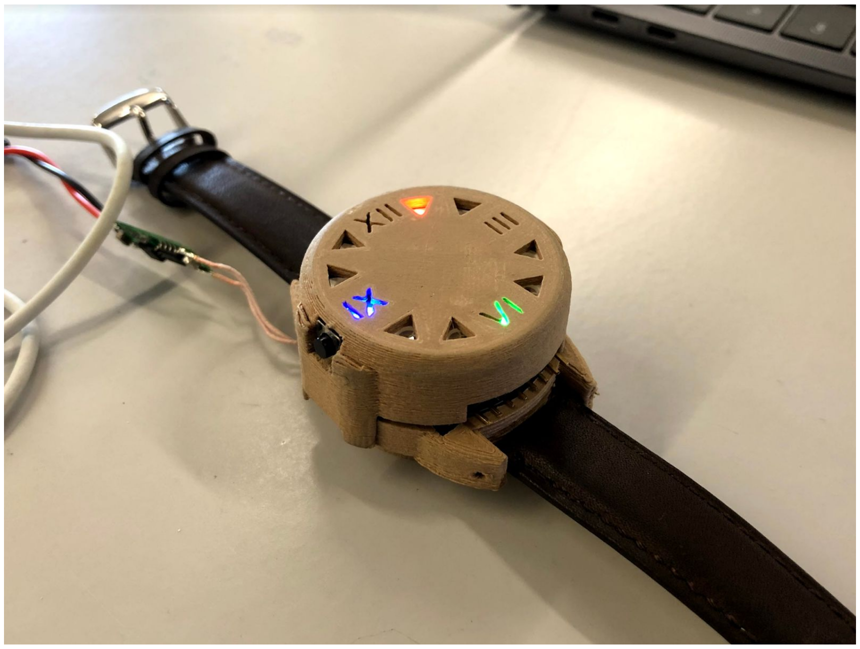


Figure 77: Timekeeping Mode

The first tested mode was standard timekeeping as described previously in Figure 77. The hour, minute, and second hands on the LED face were seen to advance and correspond with the current time.

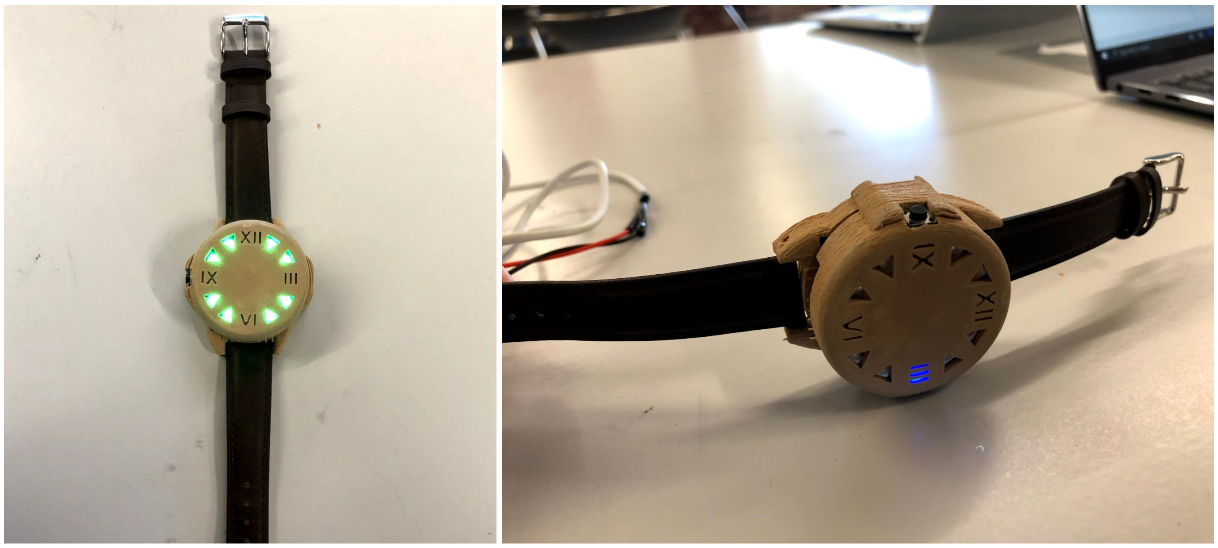


Figure 78: Motion Calibration Mode

In calibration mode, the LED face was observed at the different orientations specified previously. In Figure 78 (left), the watch is resting on a flat surface while facing up, resulting in the expected all-green indicator for level detection. In the right image, the watch is resting on its side with the 3 o'clock axis facing downward, resulting in the expected blue indicator for the directional tilt.

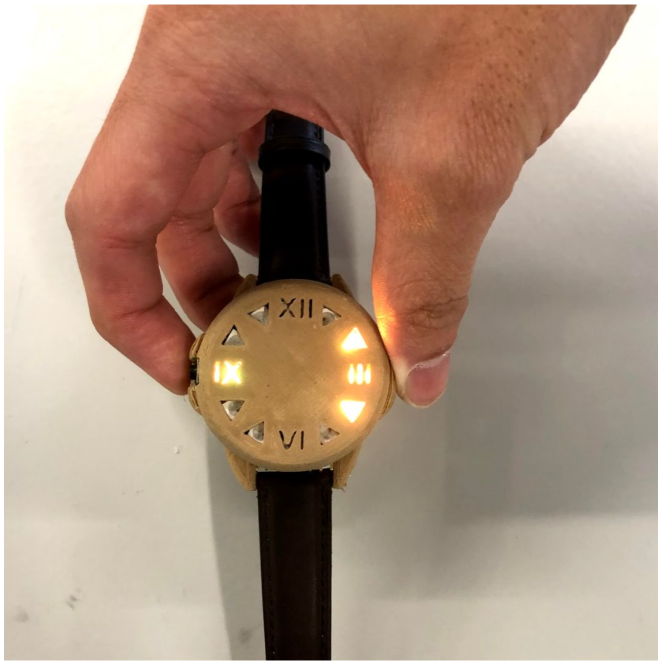


Figure 79: Gesture Control Mode

In gesture control mode, the watch was worn while performing various gestures as defined previously in Figures 77-79. For each successfully demonstrated gesture motion, the corresponding device action was executed on the paired device, accompanied with vibration feedback and an LED indicator for the action. For instance, the right-swipe gesture resulted in the paired computer advancing to the next slide in an open slideshow presentation and the corresponding "right arrow" symbol pictured in Figure 79.

(See video in Appendix for full demonstration)

Final Results

The following list outlines the major subsystems (labeled 1-4) and functional components (labeled a-d) that we intended to implement and demonstrate in the final product of our project, along with the degree of completion that we actually achieved in implementing them.

1.) Communication subsystem
   a.) Bluetooth connectivity: we successfully paired the watch with a PC and executed remote control of its applications, demonstrating a complete Bluetooth interface
2.) Sensor subsystem
   a.) Gyroscope sensing: we captured and processed full 6-axis orientation detection and visually demonstrated this through the watch's calibration mode

    b.) Accelerometer sensing: we captured and processed full 6-axis acceleration detection through at least 2 directionally distinct gestures
    c.) Gesture interpretation controls: we implemented at least 2 distinct gesture actions that were intuitively activated by test users without significant false positives
    d.) Pushbutton control: we allowed direct toggling between software modes through a basic switch input device
3.) Feedback/actuator subsystem
    a.) Vibration: we completely implemented haptic feedback by triggering a vibration motor through software control as needed
    b.) LED array feedback: we completely implemented continuous software control of a 12-segment LED array as needed
    c.) LED timekeeping: we successfully synced said LED control to an internal clock
4.) Power subsystem
    a.) Battery power/protection: we successfully powered all of the aforementioned systems with a 3.7V battery and simultaneously guarded from under/overvoltage
    b.) Wireless charging: we successfully recharged the battery as needed through a wireless inductive charging interface

Overall, we sufficiently met all of the expectations we set for ourselves at the beginning of this project and even surpassed our basic requirements in gesture and motion control, achieving several bonus functionalities.

Costs

The estimated total cost of our project, considering all components, materials, services, and tools used, amounts to a little over $330 for the single unit that we produced. Notably extraneous contributions to this cost include the MSP430's header board used in prototyping and testing ($25), the excessive supply of 3D printing filament ($20), as well as the extended cost of labor for manually mounting the PCB with small components that were difficult to work with ($180).

If the watch were to be upscaled to mass production, many of the overhead costs of manufacturing could be streamlined. Increasing batches to 100,000 units could reduce the cost of bulk orders for SMD chip components to as little as 1/90th of the price per unit, stripping almost $15 in part costs from each unit. Also, each 1kg spool of plastic filament ordered could support the housing for approximately 70 more units, which also virtually eliminates the $20 cost of 3D printing for each unit. The significant amount of labor in manual PCB assembly could also be easily replaced by machine manufacturing, as the common SMD chip packages we used can be automatically placed and soldered in large batches, significantly reducing the $180 cost of PCB assembly. Considering these changes, along with the other miscellaneous inefficiencies of our design process, it is entirely possible that costs could be brought under $100 per watch.

(See Figure 1A in the Appendix for the detailed cost spreadsheet)

Future Work

There are many improvements that we wish to make to the project if we were to continue it further. Many of them involve quality of life changes that make debugging more efficient or the PCB more compact. Given another semester, we would have explored how to add the push button as a surface mount device to the side of the PCB instead of having a free wire for it. A magnetic docking could be added to the watch and inductive charger chassis to enable more reliable charging. Smaller package resistors and capacitors could have been used to potentially shrink the PCB further. Another change that could potentially reduce the PCB size is to increase it to 4 layers from the current 2 layers used. The top chassis could have been changed to have a screw hole and allow for a more secure fit. A shunt on the JTAG helper board that we created also would have been of great help when it came to faster debugging. There are also many new hardware features that we wish to implement such as an LCD screen or a GPS for more versatile timekeeping. The final things that we wished could be improved upon comes from the software end. We wished to add a reaction time game that could be implemented via Bluetooth and have a GUI from the computer's end or a mouse mode that used the gyroscope/accelerometer to move the mouse and the pushbutton to perform a click.

The main difficulties during this process were waiting for the components. The one advisable thing to do is order things early and often. Do not wait for the next order to order something. Batteries and mosfets can take weeks to arrive and many components could be missing in action. Always get frequent help on the schematics as well, as there are many vital components that could be omitted. And finally, the last piece of advice to future students is to not procrastinate. ECE capstone projects are very intensive and are very fast paced given only one semester.

References

[1] “MSP430 Ultra-Low-Power MCUs | Overview | Microcontrollers (MCU) | TI.com.” [Online]. Available: http://www.ti.com/microcontrollers/msp430-ultra-low-power-mcus/overview.html. [Accessed: 24-Sep-2019].

[2] “Specification of the Bluetooth System”. [Online]. Available: http://grouper.ieee.org/groups/802/15/Bluetooth/core_10_b.pdf. [Accessed: 24-Sep-2019].

[3] “smarchWatch”. [Online]. Available: https://github.com/S-March/smarchWatch_PUBLIC. [Accessed: 24-Sep-2019]

[4] “SmartBondTM DA14682 and DA14683,” Dialog Semiconductor, 28-May-2018. [Online]. Available: https://www.dialog-semiconductor.com/products/connectivity/bluetooth-low-energy/smartbond-da14682-and-da14683. [Accessed: 24-Sep-2019].

[5] “Printed Circuit Board Manufacturer & PCB Assembly | Advanced Circuits.” [Online]. Available: https://www.4pcb.com/. [Accessed: 24-Sep-2019].

[6] “UVa Course Catalog - Complete Catalog for the Electrical and Computer Engineering Department (Unofficial, Lou’s List).” [Online]. Available: https://rabi.phys.virginia.edu/mySIS/CC2/ECE.html. [Accessed: 24-Sep-2019].

[7] “Program and Data Representation: CS 2150 Specific Content.” [Online]. Available: https://aaronbloomfield.github.io/pdr/uva/index.html. [Accessed: 24-Sep-2019].

[8] “ECE 3430 - Introduction to Embedded Computer Systems at the University of Virginia | Coursicle UVA.” [Online]. Available: https://www.coursicle.com/virginia/courses/ECE/3430/. [Accessed: 24-Sep-2019].

[9] “cplusplus.com - The C++ Resources Network.” [Online]. Available: http://www.cplusplus.com/. [Accessed: 16-Dec-2019].

[10]“SMD - Surface Mount Device,” *Eurocircuits*.

[11] “Small Outline Transistor (SOT) Package Types.” [Online]. Available: https://eesemi.com/sot-types.htm. [Accessed: 16-Dec-2019].

[12] “DigiKey Electronics - Electronic Components Distributor.” [Online]. Available: https://www.digikey.com/. [Accessed: 24-Sep-2019].

[13]“Mouser Electronics - Electronic Components Distributor.” [Online]. Available: https://www.mouser.com/. [Accessed: 16-Dec-2019].

[14] “MISRA - The Motor Industry Software Reliability Association.” [Online]. Available: https://www.misra.org.uk/. [Accessed: 16-Dec-2019].

[18] “CCSTUDIO Code Composer Studio (CCS) Integrated Development Environment (IDE) | TI.com.” [Online]. Available: http://www.ti.com/tool/CCSTUDIO. [Accessed: 24-Sep-2019].

[19] “Multisim Live Online Circuit Simulator,” NI Multisim Live. [Online]. Available: https://www.multisim.com/. [Accessed: 24-Sep-2019].

[20] “Ultiboard - National Instruments.” [Online]. Available: http://www.ni.com/en-us/shop/select/ultiboard. [Accessed: 24-Sep-2019].

[21] “Inventor | Mechanical Design And 3D CAD Software | Autodesk.” [Online]. Available: https://www.autodesk.com/products/inventor/overview. [Accessed: 26-Sep-2019].

[22] “Ultimaker Cura: Powerful, easy-to-use 3D printing software.” [Online]. Available: https://ultimaker.com/software/ultimaker-cura. [Accessed: 26-Sep-2019].

[23] “RealTerm Terminal Software.” [Online]. Available: https://realterm.sourceforge.io/. [Accessed: 16-Dec-2019].

[24] “AutoIt”. [Online]. Available: https://www.autoitscript.com/site/. [Accessed: 16-Dec-2019].

[25] A. M. Davydov, “(71) Applicant: Apple Inc., Cupertino, CA (US),” p. 23.

[26] W. Kienzle and K. P. Hinckley, “Smart ring,” US9582076B2, 28-Feb-2017.

[27] A. M. DAVYDOV, “Wristband device input using wrist movement,” US20160299570A1, 13-Oct-2016.

[28] “RN42/RN42N Class 2 Bluetooth Module with EDR Support.” [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/50002328A.pdf. [Accessed: 24-Sep-2019].

[29] “MPU-6050 TDK InvenSense | Sensors, Transducers | DigiKey” [Online]. Available: https://www.digikey.com/product-detail/en/tdk-invensense/MPU-6050/1428-1007-2-ND/403800 9l. [Accessed: 24-Sep-2019].

[30] “ROB-08449 SparkFun Electronics | Motors, Solenoids, Driver Boards/Modules | DigiKey” [Online]. Available: https://cdn.sparkfun.com/datasheets/Robotics/B1034.FL45-00-015.pdf. [Accessed: 15-Dec-2019].

[31] “NeoPixel Ring - 12 x 5050 RGB LED with Integrated Drivers” [Online]. Available: https://www.adafruit.com/product/1643. [Accessed: 15-Dec-2019].

[32] “Wireless Charging Module 5V/300mA-DFRobot” [Online]. Available: https://www.dfrobot.com/product-1283.html?gclid=Cj0KCQjwoKzsBRC5ARIsAITcwXEspswIvyiB1Q6sAQh-SAp9RjlocUqFxZQUHykKQ3fFlpWXdAISqi0aAjKLEALw_wcB. [Accessed: 15-Dec-2019].

[33] “RJD Series Rechargeable Li-Ion Batteries | Illinois Capacitor” [Online]. Available: http://products.illinoiscapacitor.com/seriesDocuments/RJD_series.pdf. [Accessed: 15-Dec-2019].

[34] “TL3305 E-Switch | Switches | DigiKey” [Online]. Available: https://sten-eswitch-13110800-production.s3.amazonaws.com/system/asset/product_line/data_sheet/213/TL3305.pdf. [Accessed: 15-Dec-2019].

[35] “S-8261 Series Battery Protection IC” [Online]. Available: https://www.ablic.com/en/doc/datasheet/battery_protection/S8261_E.pdf. [Accessed: 15-Dec-2019].

[36] “Resistor Sizes and Packages” [Online]. Available: http://www.resistorguide.com/resistor-sizes-and-packages. [Accessed: 15-Dec-2019].

[37] “Small Outline Transistor (SOT) Package” [Online]. Available: https://eesemi.com/sot.htm. [Accessed: 15-Dec-2019].

[38] “IEEE 1625-2004 - IEEE Standard for Rechargeable Batteries for Portable Computing.” [Online]. Available: https://standards.ieee.org/standard/1625-2004.html. [Accessed: 24-Sep-2019].

[39] “ECE 3430 - Introduction to Embedded Computer Systems at the University of Virginia | Coursicle UVA.” [Online]. Available: https://www.coursicle.com/virginia/courses/ECE/3430/. [Accessed: 24-Sep-2019].

[41] “ECS-.327-8-14 ECS Inc. | Crystals, Oscillators, Resonators | DigiKey.” [Online]. Available: https://www.digikey.com/product-detail/en/ecs-inc/ECS-.327-8-14/X803-ND/109993. [Accessed: 16-Dec-2019].

[42] “GRM033R61E472MA12D Murata Electronics | Capacitors | DigiKey.” [Online]. Available: https://www.digikey.com/product-detail/en/murata-electronics/GRM033R61E472MA12D/490-9994-1-ND/5026313. [Accessed: 16-Dec-2019].

[43] “RC0603JR-074K7L Yageo | Resistors | DigiKey.” [Online]. Available: https://www.digikey.com/product-detail/en/yageo/RC0603JR-074K7L/311-4.7KGRCT-ND/729732. [Accessed: 16-Dec-2019].

[44] “RNCP Series Resistors - Stackpole Electronics | DigiKey.” [Online]. Available: https://www.digikey.com/en/product-highlight/s/stackpole-electronics/rncp-series-high-power-thin-film-resistors. [Accessed: 16-Dec-2019].

[45] “S-8261AAJMD-G2JT2G ABLIC U.S.A. Inc. | Integrated Circuits (ICs) | DigiKey.” [Online]. Available: https://www.digikey.com/product-detail/en/ablic-u-s-a-inc/S-8261AAJMD-G2JT2G/728-1034-1-ND/1628437. [Accessed: 16-Dec-2019].

[46] J. Valdez and J. Becker, “Understanding the I2C Bus,” p. 8, 2015.

Appendix

Item	Quantity	Total Price
MSP430F1611	1	$11
MSP430F1611 Header Board	1	$25
Gyroscope/Accelerometer	1	$6
Vibration Motor	1	$3
12 x 5050 RGB LED Ring	1	$8
Lithium-ion Battery	1	$10
Wireless Charging Module 5V/300mA	1	$7
Chip Resistors/Capacitors/Transistors/etc.	-	$15
Pushbutton	1	$0.5
PLA Plastic	1	$20
PCB Manufacture from Advanced Circuits	1	$33
Watch Band	1	$13
Watch Pin	1	$6
3W Mounting	-	$60 * 3 Hours = $180
Total Price		$337.5

Figure 1A: Cost Estimation

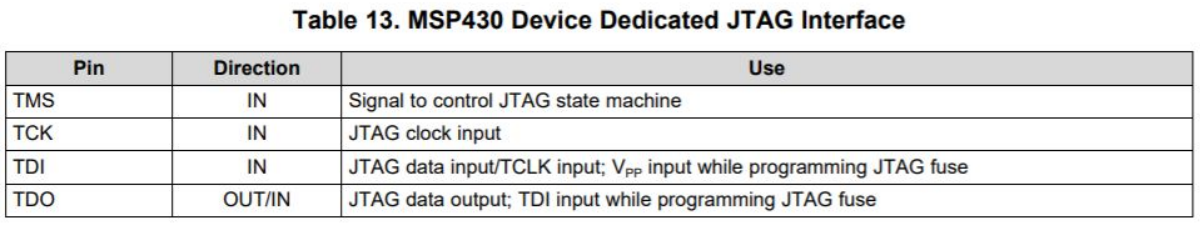


Table 13. MSP430 Device Dedicated JTAG Interface

Pin	Direction	Use
TMS	IN	Signal to control JTAG state machine
TCK	IN	JTAG clock input
TDI	IN	JTAG data input/TCLK input; V_{PP} input while programming JTAG fuse
TDO	OUT/IN	JTAG data output; TDI input while programming JTAG fuse

Figure 80: JTAG Interface for MSP430