[[1]](#footnote-1) ­­

Leaf: A Wearable Social Networking Device

Marco Chiang, CSE; Chris Garry, CSE; Steven Tso, CSE; and Aaron Tye, EE

*Abstract*—The goal of the Leaf bracelet is to enable the exchange of social media contact information by establishing connection requests with a simple handshake. Currently, contact information is exchanged on paper using resumes or business cards. The Environmental Protection Agency reports that 28% of total municipal solid waste generation in­­ the United States comes from paper alone. Leaf aims to reduce MSW generation as well as demonstrate the capabilities of social media as an effective medium for personal and professional network development.



Fig. 1. Leaf bracelet design concept.

# INT­RODUCTION

S

ocial networking is one of the most important aspects of professional and personal development. People frequently attend events to meet with potential clients, employees, business partners, and friends. Building connections at these events typically involves an exchange of contact information with a mobile phone or in the form of a resume or business card. However, with the increasing adoption of online social networks, social media is becoming a popular medium for people to connect. A report by Jobvite Inc. states that 94% of recruiters use or plan to use social media in their recruitment efforts and 78% of recruiters have hired through social media [1]. Unlike a social media profile, phone numbers can be difficult to manage and remember, and resumes and business cards need to eventually be disposed of. The Environmental Protection Agency reports that the total municipal solid waste (MSW) generation in 2011 was 250 million tons, 28% of which comes from paper, making it the largest contributor to MSW [2]. This rise in social media and proliferation of paper waste has created an opportunity to innovate the conventional means of personal and professional networking.

Leaf is an electronic bracelet designed to address these issues and remove the burden of exchanging and managing contact information from the user (figure 1). A Leaf user will wear the device on the wrist in a watch-like fashion. When two Leaf users shake hands, each bracelet triggers a connection request with the acquaintance on their social network of choice, the settings of which are controlled through the Leaf mobile application, providing for a more seamless, environmentally friendly way to connect.

TABLE I

Leaf Specifications

|  |  |
| --- | --- |
| Specification | Value |
| Weight | <20g |
| Height | <1.5cm |
| Length | <26cm |
| Width | <2cm |
| Battery Life | >7 hours |

Target Price <$50

# Design

## Overview

Leaf is being designed to eliminate the need to store and exchange contact information by allowing users to make social media connections with a simple handshake. This will be accomplished by implementing a wrist bracelet that monitors movement data using an accelerometer. When the bracelet detects a handshake in the accelerometer data using the on-board handshake detection algorithm, the portion of the movement data encapsulating the handshake is sent to the user’s mobile phone via Bluetooth, which then pushes the data to the back-end server of the Leaf mobile application. The back-end server will store the accelerometer data and timestamp followed by executing the back-end matching algorithm to determine which two Leaf users shook hands at the specified timestamp by comparing handshake data. Once the user’s identities are confirmed, the server will push basic information (name) of the respective persons to the mobile devices of each user involved in the handshake, where the user can then approve or disapprove the connection request. If a request is approved, the app will send a connection request on the social network defined in the mobile app settings. The connection request must be approved on both participating devices in order for the application to initiate it on the appropriate social network.

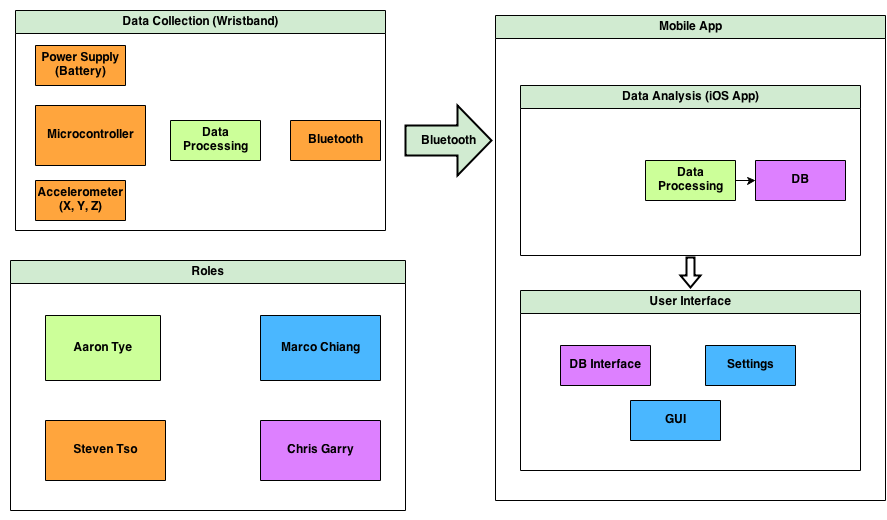


Fig. 2. Block diagram for Leaf. The Leaf bracelet will have an embedded microcontroller that controls the device peripherals and handshake detection algorithm. The device peripherals consist of an accelerometer to gather movement data, a Bluetooth module to communicate this data to the mobile phone, and a battery to power the device. The mobile app will process and store this data on the back-end server. The mobile app interface will display pending connection requests from the server as well as allow users to configure social network and auto-approval settings.

The use of the Leaf bracelet will help reduce the need for paper-based information exchange, reducing contributions to MSW. Further, Leaf will allow the ability to make connections on the most popular social networks, both personal and professional, such as LinkedIn and Facebook. There are currently no devices on the market today, to the best of our knowledge, which are designed to tackle the problems of social media integration and MSW generation from paper resumes and business cards. A block diagram for the Leaf device can be seen in figure 2 above.

TABLE II

STML32L Select Specifications [3]

|  |  |
| --- | --- |
| Specification | Value |
| Architecture  Flash Memory  RAM  EEPROM | Harvard  256 KB  32 KB  8 KB |
| Peripherals  Communication | 9  USART, USB, SPI |
| Clock  Power | 32 kHz - 32 MHz max  187 μA/MHz Run mode  8.6 μA Low-power |

## Block 1: Data Collection

The data collection block consists of all hardware (and associated software) mounted on the bracelet to acquire the movement data and communicate it to the Leaf mobile application. This hardware module consists of a microcontroller, data transceiver, accelerometer, status indicator, handshake detection algorithm, and power supply. In choosing these hardware components, the selection criteria were low power consumption, price, and ease of use. The testing of the block consists of a development board setup with communication of accelerometer and timestamp data from the bracelet to a mobile phone. Leaf is being developed in coordination with the CS Sensors Lab at the University of Massachusetts Amherst. The Sensors Lab is providing assistance in choosing hardware components and developing a finished product with a small form factor.

**Microcontroller**

It was decided that the central processor chosen for the bracelet should be a microcontroller solution that provides high access to peripherals, a small form-factor, and low cost. In deciding the microcontroller architecture, it was found that an ARM-core processor would be appropriate since the RISC architecture allows for ultra-low power consumption and execution efficiency. Other considerations were clock speed compatible with the peripherals and enough RAM and non-volatile flash memory to process the data we plan to work with at any given point of chip operation. All of these criteria were met by the STM32L microprocessor shown, which is an ultra-low-power 32-bit MCU ARM-based Cortex-M3 [3]. The specs for the STML32L microcontroller can be seen in table 2.

To program the microcontroller, we are using a development board with an onboard USB/JTAG interface to the chip. The IAR embedded workbench is being used to develop and program the microcontroller firmware due to its frequent use in the industry and its support for the chip [4]. During the final stages of development, there will be a JTAG interface directly to the chip for programming.

**Data Transceiver**

Transmission and receiving of data from the device is done wirelessly with Bluetooth. Looking at different solutions, Bluetooth was a sensible option due to its wide consumer appeal and its market margin over embedded devices. An alternative solution was Zigbee, but the widespread use of Bluetooth in wearable technologies provides a larger support community [5]. Different Bluetooth versions were considered, such as the 3.0 and 4.0, but since Bluetooth 4.0 is still relatively new compared to 3.0 and may not be implemented in all devices, we again opted for the solution with a large support community and compatibility, which was version 3.0.

To connect to the Bluetooth device, there will be a “discoverable” button on the Leaf device to allow incoming pairing requests. For the Bluetooth firmware, there were two different options.

A custom software stack could be implemented on the microcontroller or a module with the stack implementation could be purchased, which is what was chosen for the device. The SPBT2632C2A from STMicroelectronics (Figure 3) was purchased and contains an implementation of a Bluetooth stack called Amp’ed RF, which uses the UART line to send and receive Bluetooth commands [6].

Security and privacy are always concerns when working with wireless technologies. This Bluetooth module offers 128-bit AES encryption for communication with the mobile phone [4], which can help prevent attacks that rely on intercepting or listening to the communication channel.

TABLE III

Bluetooth 3.0 Select Specifications [4]

|  |  |
| --- | --- |
| Specification | Value |
| Compatibility | ST micro Cortex-M3 microprocessor up to 72 MHz (256 kb Flash, 48 kb RAM memory) |
| Security  Operating Voltage  Size | 128-bit encryption  2.5 V typical  11.6 x 13.5 x 2.9 mm |
|  |  |

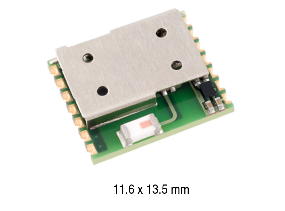


Fig. 3. Bluetooth 3.0 module.

**Accelerometer**

The role of the accelerometer is to acquire wrist movement data in 3D space. Several options for the accelerometer were available: a 3-axis, which consists of solely an accelerometer, a 6-axis, which has an accelerometer and gyroscope, and a 9-axis, which has an accelerometer, gyroscope, and compass. The 3-axis was chosen for the device as the data in more than three dimensions would introduce unnecessary complexity, and thus power consumption, into the device. The model of accelerometer chosen was the Invensense MPU-6050, as Invensense is an established designer of quality inertial measurement chips at low cost. The chip interfaces using Inter-Integrated Circuit (I2C), a communication bus designed to interface low-speed peripherals with the processor. IC2 comes standard with the STML32L microcontroller.

**Status Indicator**

    One of the goals of Leaf is to be as unobtrusive as possible. The device does not need to provide much feedback to the user as it acts as an extension for the mobile app. Thus, it was decided that a small, multicolored light-emitting diode (LED) would be ample to communicate device state to the user. The specifications of this component have not been decided.

**Firmware State Machine**

In the simplest sense, the wristband operates in three states: initialize, pair, and poll. The machine starts at initialize to get the accelerometer and Bluetooth running when powered on. After, the device moves into the pairing state, where it will keep checking for a paired device. Until a pair is detected, the device will stay in this state. There will be a visual indicator conveying whether or not the device is in this stage. After it connects to a mobile phone, the device will enter the polling stage where it will process accelerometer data to detect a handshake and subsequently send this information via Bluetooth to the paired mobile phone. A preliminary state machine can be found in figure 4. The finalized state machine will have many more layers. A picture of the hardware prototype can be found in the appendix.

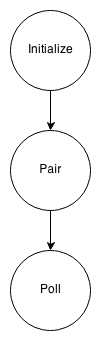


Fig. 4. Preliminary state machine diagram for Leaf device. Upon device boot, initialization of peripherals will take place. After initialization, the device will remain in the Pair state until a mobile device has been paired with the bracelet. After the pair, the bracelet will poll the accelerometer for handshake data, which will be sent to the mobile device.

**Handshake Detection Algorithm**

The purpose of this algorithm is to detect a handshake given signals from an accelerometer worn on the wrist. Once a handshake has been detected by the microcontroller, the bracelet will send the accelerometer data segment of the handshake with timestamps to the user’s phone via Bluetooth. This algorithm was written in MATLAB and will be ported to C for the microcontroller using MATLABs built-in cross compilation functionality.

TABLE IV

Accelerometer Select Specifications [7]

|  |  |
| --- | --- |
| Specification | Value |
| Protocols | IC2 |
| Operating Voltage  Size | 2.1 to 3.6 V  4x4x0.9 mm |
|  |  |

For testing, handshake accelerometer data was gathered with an iPhone 5 through a custom developed mobile application design just for proof of concept. This application samples at a rate of 100 ms and outputs data in gravitational force. Figure 5 represents what a typical handshake looks like.

The handshake detection algorithm operates on 12 ms wide segments of real-time accelerometer data. After curating and analyzing various handshake data, it was found that handshakes consistently begin with a sharp, downward motion in the x-axis. Thus, a heuristic was developed to indicate whether or not the algorithm thinks a handshake is about to take place to conserve power spent on unnecessary computations. The heuristic polls the x-axis data to determine if the gravitational force is indicative of the initial downward motion of a handshake. If the initial downward motion falls below the heuristic threshold of 0.75 gravitational units, the algorithm will begin to further analyze the segment to determine if a handshake is taking place. After this stage, several features that are characteristic of a handshake are extracted to classify the data.

The first feature involves the Discrete Fourier Transform (DFT) of the accelerometer data. The DFT of a segment *x* of length *n* is another segment *y* of length *n:*

where *ω* is a complex *n*th root of unity:

This notation uses *i* for the imaginary unit, and *p* and *j* for indices that run from 0 to *n*–1. The indices *p*+1 and *j*+1 run from 1 to *n*, corresponding to ranges associated with MATLAB vectors.

The DFT returns a segment indicating the component frequencies in the accelerometer data (i.e. the frequency of any oscillations found in the accelerometer segment). From testing, it was found that a typical handshake has a large component frequency between 2 and 4.5 Hz. This feature will provide us with the component frequencies of a segment to determine if it contains oscillations between the 2 and 4.5 Hz.

Following the computation of the DFT, the second feature, called the Crest Factor (CF), is extracted [8]. The CF is a scalar indication of the intensity of a peak in a signal by taking the ratio of the peak amplitude of a signal to the average RMS value of the signal. The crest ratio is computed by:

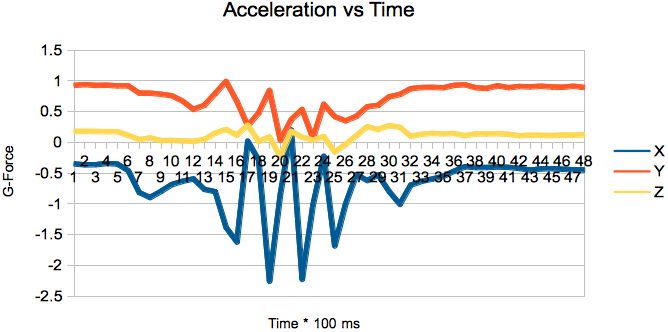


Fig. 5. Accelerometer output for a handshake.

The CF is a good feature to extract from the accelerometer data because it was found during testing that handshakes typically demonstrate a CF between 1.4 and 2.5 on the x-axis. Plotting the CF with the peak frequency in the DFT reveals a clustering pattern for handshakes. Thus, the algorithm will look for segments whose CF and DFT values fall within the cluster below, bounded by a peak frequency between 2 and 4.5 Hz and a CF between 1.4 and 2.5.

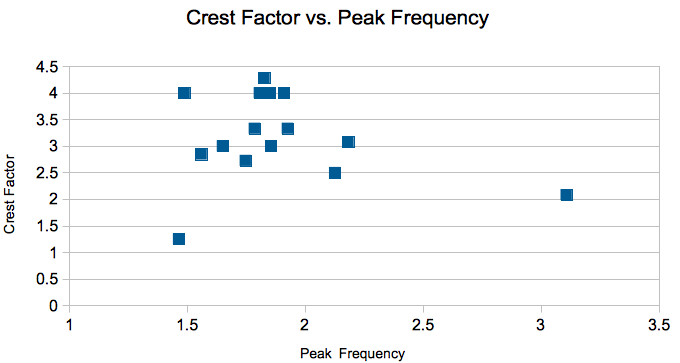


Fig. 6. A plot of Crest Factor vs. Peak Frequency reveals a cluster of points that belong to a handshake.

The last feature extracted is a count of the number of oscillations on the y-axis of the segment. It was found during testing that handshakes demonstrate activity in the y-axis where oscillations correspond with the number of shakes made. A threshold for the number of oscillations was set to three. If the number of crossings of the y-axis is greater than three and has amplitude greater than 0.4 gravitational units, this feature will return true. These three features combined result in the algorithm described in figure 7.

|  |
| --- |
| **input**: XYZ accelerometer data segment |
| **while** device on: |
| **if** data segment contains sudden downward Y motion:  compute peak frequency by Discrete Fourier Transform;  compute crest ratio;  compute number of Y oscillations; |
| **if** peak frequency, crest ratio, oscillations match criteria:  handshake segment++; |
| **if** handshake segment > 3  **return** handshake true; |
| **return** handshake false |

Fig. 7. Leaf bracelet handshake detection algorithm.

The algorithm has been tested on a small dataset with 100% accuracy under false positive potentials such as waving hands. Future testing will consist of a more varied dataset that is more realistic than the test scenario. Existing algorithms utilize advanced machine learning methods, which are computationally intensive and not suitable to execute on an embedded device [9].

**Power Supply**

The battery model has not been specified up to this point. The model will be chosen after the hardware prototype is complete and efficiencies have been made to optimize power consumption. Figure 8 contains a preliminary power consumption plan that will guide in selecting the appropriate battery.

## Block 2: Mobile App

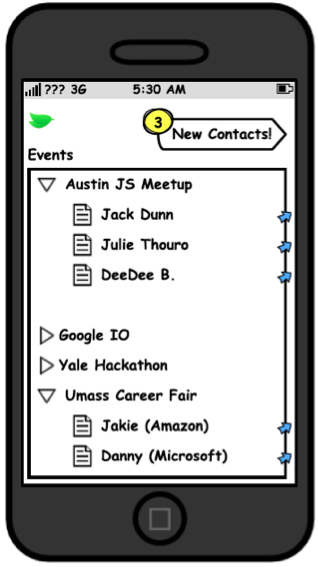
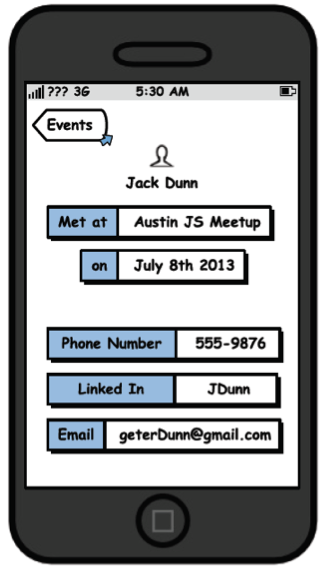
Leaf mobile is intended to be a simple graphical user interface for users to access and maintain their contacts. The mobile app will act as a mediator, receiving information from the bracelet, sending it to a backend server, and then getting a response: which two users actually shook hands. Once the bracelets detect a handshake, they send the raw accelerometer data in a batch along with a timestamp to the mobile app. When the server receives data from the mobile app, the back-end matching algorithm will first look for handshakes that have occurred at the same time. If multiple handshakes occurred at the given timestamp, a comparison of the physical accelerometer data will take place to determine a match. One option considered for matching accelerometer data was cross correlation, which is a measurement of the similarity of two waveforms defined by

where and are the two waveforms and denotes the complex conjugate of . When a match is confirmed, the server will respond with the exchanged contact and the app will notify the user that a new contact has been established. The user then has the option to request contact information from another user that they’ve physically met.

The goal is to develop the mobile app on the Android platform, as it is open-source and easily accessible. An android prototype version has been created that utilizes the Pebble smart watch as the bracelet. The Pebble smart watch is a wrist device that interfaces with your mobile phone to present useful data and notifications from your mobile phone to your wrist. The smart watch has an accelerometer, which made it a good candidate for prototyping. The prototype utilizes two Pebble smart watches worn on the right hand of two users, accessing on-board accelerometer data to detect when a handshake occurs. As this prototype was done prior to the development of the handshake algorithm in figure 7, the algorithm used in this instance was very basic and simply implemented a threshold flag where a handshake was registered if the accelerometer read values greater than threshold. The mobile application then exchanges contact information. When the two users shake hands, the Pebble watches detect the handshake and send a confirmation to their mobile phones. Once the mobile application receives the confirmation it sends a timestamp to the backend server created in Node.js hosted on Amazon Web Services (AWS). The Node.js application looks for two timestamps within a one hundred millisecond window and matches those two as a handshake. It then responds to the app with the other contact’s full name to be displayed on the mobile app. Mockup designs for the mobile app user interface are presented below.

|  |  |  |
| --- | --- | --- |
|  | Average Case (mAh) | Worst Case (mAh) |
| Microcontroller | 3 | 6 |
| Accelerometer | 2 | 4 |
| Bluetooth | 10 | 18 |
| TOTAL | 15 | 28 |

Fig. 8. Preliminary power budget.

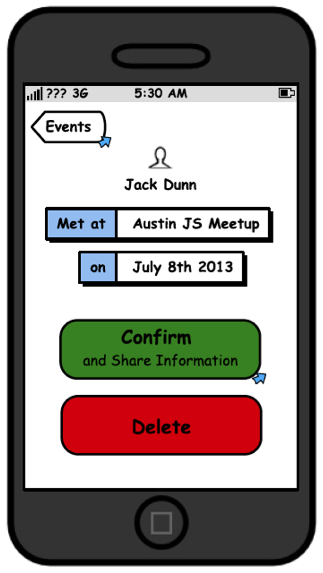
 

Fig. 9. Leaf mobile user interface concepts.

# Project Management

|  |  |
| --- | --- |
| **Specification** | **Completion** |
| Sensor/Microcontroller Interface | 100% |
| Microcontroller/Bluetooth Communication | 100% |
| Transmit Sensor Data to Mobile Device | 100% |
| Mobile GUI mockup | 100% |
| Handshake Detection Algorithm Prototype | 100% |

Fig. 10. MDR deliverable specification and completion

**Team Responsibilities**

Each team member’s individual roles were first assigned after the team broke down the project requirements. The team lead, Marco Chiang, assessed each member’s strengths and distributed responsibilities to balance each member’s workload.

Steven Tso has experience with microcontrollers and thus took the majority of the responsibility for the hardware. Marco ensured that Steven had all the necessary hardware parts, support, and specifications to complete his part. This included reaching out to the UMass CS Sensors Lab who has thus provided us with development boards and graduate student consulting. To date, Steven has successfully demonstrated functional hardware by sending accelerometer data to his mobile phone from the microcontroller over Bluetooth.

Aaron Tye expressed interest in signal processing, so his responsibilities include gathering raw accelerometer data and using all the data to create a classification algorithm to detect a handshake. Our project’s success relies heavily on how accurately our algorithm can detect when a user engages in a handshake. Marco ensured that Aaron had all the necessary tools to collect raw accelerometer data by developing an iPhone application that accesses the phones accelerometer and emails the data in excel format. To date, Aaron has gathered handshake data and has written a handshake detection algorithm in MATLAB.

    Chris Garry has expressed interest in both hardware and software, although his role has been altered several times due to unforeseen complications. Originally his responsibilities included website management, database implementation, and Bluetooth interfacing with microcontroller. Marco met with Chris several times to identify the source of his complications and adjusted for new responsibilities. Chris has experience in both hardware and software and now acts as a supporting role helping Steven with hardware, assisting Aaron with the handshake algorithm, and also consulting Marco with his responsibilities.

    Marco Chiang has expressed interest in mobile development and, besides being team lead, has taken the role of developing the mobile application that will eventually be paired with the hardware device that Steven is developing. To demonstrate the use and application for the Leaf bracelet and mobile application, Marco used the Pebble smart watch and its development platform to create a prototype. The prototype was demonstrated in front of over one thousand people, the direct target market, and was widely received among those who witnessed our product in action. The technology and product was awarded as one of the top 6 teams who competed in the Yale Hackathon from November 8th-9th. A Gannt chart for the project completion milestones can be found in figure 11 of the appendix.

**Team Communication**

Marco schedules at least two to three meetings every week. Team Leaf often meets twice a week to share updates/progress and discuss implementation methods. Our weekly team meetings are scheduled for Mondays and meetings with our Advisor, Professor Siqueira, are on Wednesdays. In the interim, we communicate through a group texting service to get real-time updates and coordinate meetings. We collaborate to share informative web links using a Facebook group page and Google drive. Occasionally, team members will meet with one another to clarify any questions or receive help and support. At this point, we’ve never had issues or blocks when meeting with our advisor, so we typically give a report of our progress and continue working on our own parts. During each team meeting, Marco starts off with an overview of his accomplishments and in turn asks each member to give an update and express any concerns or roadblocks. We then discuss how to address and solve these issues at the meeting and then establish what our goals for our next meeting will be.

# Conclusion

The current state of the Leaf device is a working hardware prototype that communicates accelerometer data over Bluetooth to a mobile app and a working handshake detection algorithm. The back-end matching algorithm, mobile app, and server are to be implemented in the future. These will then need to be integrated and sized down to a bracelet form. The current state of the project is a result of great team work ethic and communication.

The use of communication tools and frequent meetings has kept the project development at a high pace. Leaf’s near-future plans involve Steven and Marco meeting with an embedded systems consultant with almost 30 years of experience in electrical and mechanical engineering systems. The goal is to determine how we can reduce the costs for manufacturing of the Leaf bracelets. Aaron’s primary goal is to load the handshake algorithm into the microprocessor to see if the device can detect handshakes. Marco will be working on interfacing the mobile application with the development board. The board will eventually be reduced to a miniature device that can easily fit in a live-strong type bracelet, which will require PCB design. Chris will then focus on implementing the backend database and define the schema.

We expect to face difficulties in achieving the desired form factor, although with the guidance and resources of the CS Sensors lab and hardware consultants, we anticipate that we will be able to overcome this obstacle.

References

1. Jobvite Inc., “2013 Social Recruiting Survey Results,” *Jobvite*, Jun. 2013; <http://web.jobvite.com/rs/jobvite/images/Jobvite_SocialRecruiting2013>. pdf.
2. Environmental Protection Agency, “Municipal Solid Waste Generation, Recycling, and Disposal in the United States: Facts and Figures for 2011” EPA, Washington D.C., 2011.
3. “STM32L15xRC Datasheet,” STMicroelectronics, [Geneva](https://www.google.com/search?client=firefox-a&hs=ZZw&rls=org.mozilla:en-US:official&q=geneva+switzerland&stick=H4sIAAAAAAAAAGOovnz8BQMDgzMHnxCnfq6-gaFpnoWxEgeIaWxpYKSllZ1spZ9flJ6Yl1mVWJKZn4fCscpITUwpLE0sKkktKr640Fo0fvJl1UJll7M7droz_lB5vQMAMNaCnGAAAAA&sa=X&ei=hE2ZUt30ItTioATai4KwDQ&ved=0CNQBEJsTKAIwFQ), http://www.st.com/web/en/resource/technical/document/datasheet/DM00048356.pdf
4. ST-IAR, IAR Systems, [online] 2013,http://www.iar.com/Products/IAR-Embedded-Workbench/ARM/Supported-devices/ST/?id=489, (Accessed: 11 December 2013).
5. Bluetooth, Mobile Telephony Market, [online] 2013, http://www.bluetooth.com/Pages/Mobile-Telephony-Market.aspx, (Accessed: 11 December 2013).
6. “SPBT2632C2A Datasheet,” STMicroelectronics, [Geneva](https://www.google.com/search?client=firefox-a&hs=ZZw&rls=org.mozilla:en-US:official&q=geneva+switzerland&stick=H4sIAAAAAAAAAGOovnz8BQMDgzMHnxCnfq6-gaFpnoWxEgeIaWxpYKSllZ1spZ9flJ6Yl1mVWJKZn4fCscpITUwpLE0sKkktKr640Fo0fvJl1UJll7M7droz_lB5vQMAMNaCnGAAAAA&sa=X&ei=hE2ZUt30ItTioATai4KwDQ&ved=0CNQBEJsTKAIwFQ), http://www.st.com/web/en/resource/technical/document/datasheet/DM00048919.pdf
7. “MPU-3000/MPU-3050 Product Specification,” San Jose, California, http://invensense.com/mems/gyro/documents/PS-MPU-3000A.pdf
8. Tutorial-Pwr Meter Calibration, Yokogawa Meters and Instruments Corporation, [online] 2013, http://www.yokogawa.com/ymi/tutorial/tm-tutorial\_wt\_08.htm, (Accessed: 11 December 2013).
9. Jiahui Wu, Gang Pan, Daqing Zhang, Guande Qi, Shijian Li. “Gesture Recognition with a 3-D Accelerometer,” in *Ubiquitous Intelligence and Computing*, 2009, pp 25-38.

Acknowledgments

We would like to acknowledge the CS Sensors Lab at UMass Amherst for providing us with an overwhelming amount of support for our project. We would also like to thank Texas Instruments for hardware components which we received free of charge. We would also like to thank those who have offered their time and consultation. Finally, we would like to thank the ECE Department at UMass Amherst for their support and guidance.

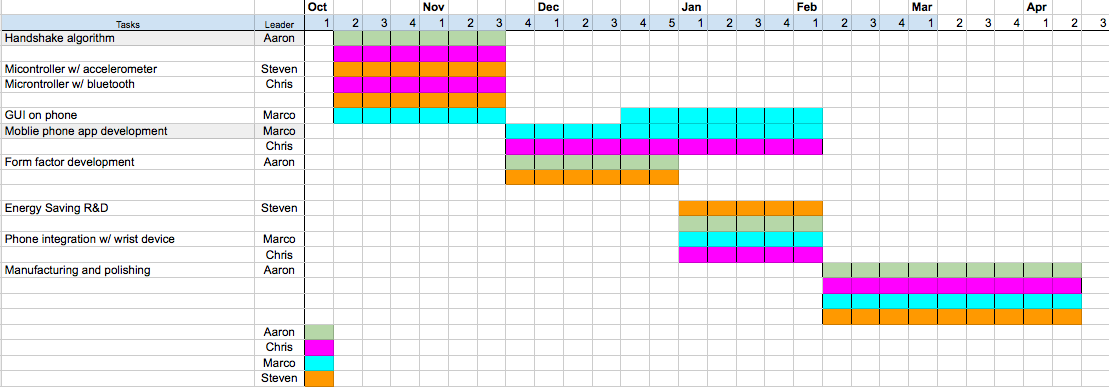
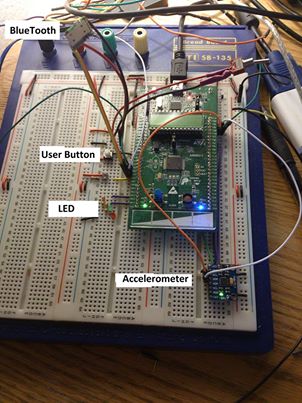


Fig. 11. Gannt chart for project timeline.

Appendix



1. M. Chiang. Author from Amherst, Ma (e-mail: mchiang@umass.edu).

   C. Garry. Author from Auburn, Ma (e-mail: cgarry@umass.edu).

   S. Tso. Author from Amherst, Ma (e-mail: stso@umass.edu).

   A. Tye. Author from Amherst, Ma (e-mail: atye@umass.edu) [↑](#footnote-ref-1)