March 10, 2016

Dr. Andrew Rawicz
Professor, School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

**RE: ENSC 440 design specification for smartBand that enhances social interaction**

Dear Dr. Rawicz,

The attached document defines the technical requirements and design standards followed to implement the prototype of smartBand. The smartBand is a device and application that would help to enhance professional connection in networking events.

The purpose of this design specification document is to demonstrate the conceptual model of our design. To justify our design, the design specifications are divided into two categories of hardware and software implementations. The hardware section emphasizes on the design and coding for each module and the software section describes the features and UI of the application.

At smartConnect, we are a team of 5 determined, passionate and focused engineering students working to provide a smart solution for people with difficulties in communicating and socializing in person. If you have any questions or concerns regarding our functional specification, please do not hesitate to contact our Chief Project Manager Sukhreet Kaur by email at ska142@sfu.ca.

Sincerely,

Gurjot Singh Atwal
CEO
smartConnect

*Enclosure: Design Specificationl for smartConnect band*

# smartConnect

**Project Team:** Gurjot Singh Atwal
Sukhreet Kaur
Rajdeep Kaur
Masih Amiri
Kevin Chang

**Contact Person:** Sukhreet Kaur
Ska142@sfu.ca

**Submitted to:** Dr. Andrew Rawicz – ENSC 440W
Steve Whitmore – ENSC 305W
School of Engineering Science
Simon Fraser University

**Issued On:** March 10, 2016
Revision: 1.0

## Abstract

The main motive of smartBand is to help people specially Businessmen, Job Seekers and Entrepreneurs to make maximum use of any “networking event” that they are attending. People attend networking events for different reasons like starting a new venture, searching for an employment opportunity or even to be on the current trends, etc. However, they all depend on one of the most common and important intent, i.e. creating relationships. In these events people have conversations with maximum 4 to 5 people and usually lose an opportunity to get in touch with many others. We will be solving this problem with the most usual gesture observed in these events i.e., hand shake. So, smartBand uses a simple handshake mechanism to create platform for people to share useful information in order to showcase their intent in the event. For example, this valuable information could be what projects a person is working on or need help in, what kind of employees or partners a businessman or an entrepreneur is looking for etc. The showcase of profiles can help people refine their search and help them to start conversations, eventually leading to mutual benefit. Therefore, smartBand will help modify the standard way of following up with someone through Business cards by providing them an active profile lookup and enabling to connect with professionals in personalized settings.

## Table of Contents

# List of Figure



# List of Tables

## Glossary

| | |
|---|---|
| EMC | Electromagnetic Compatibility |
| MPU | Microprocessor Unit |
| IC | Integrated Circuit |
| NFC | Near Field Communication |
| ASIC | Application Specific Integrated Circuit |
| App | Mobile Application |

# 1. Introduction

## 1.1. Scope

The design specification document gives a detailed description on the working of smartBand. Starting from the functional specifications, we have included the functionality for each of the sub modules, by keeping a focus on safety and efficient use of resources, and also provided the details showing the fulfillment of the requirements presented in the previous phase of the project. For the simplification of the functionality of hardware components and software application we have included system diagrams, class diagrams and user case drawings. Overall, this document focuses on the prototype of smartBand and is a good outline of the final product that we will be representing in the demonstration of the project.

## 1.2. Intended Audience

Our product serves all those individuals who want to connect with other businessmen, employers and people of their interest at the networking events in order to expand their business or social network. The product helps the business professionals seeking to hire new employees and employment seekers who would like to share profiles with each other's in order to save time looking for useful information about the other party. This document helps the production team to develop the final product for all these people by taking in consideration that all the requirements are followed.

## 1.3. Background

At smartConnect we are designing a gadget called smartBand that helps people to share each other's profile through a quick handshake, enabling them to expand their professional connections. The product includes a transmitter wristband with an app associated to it through user's phone and requires two types of communications: NFC between two bands and Bluetooth communication between the band and the app installed on the phone. When two users shake hands, the bands will interact via NFC and will share the information stored on the bands. Once the band receives data from the other band, it will communicate that information with smartConnect's app on the user's phone, displaying other user's profile. Therefore, by employing a hand shake gesture, smartBand will present useful information about the other user making it easier to have a look at other person's profile and create a connection for future follow ups.

# 2. System Design

## 2.1. System Design

Figure 1 below describes the high level working of the smartBand and also provides an overview on the final model of the system. Not emphasizing on the technical details of the interactions between the components used, the figure depicts the data transfers between main modules. As shown below, smartBand uses two main interactions: Near Field Communication (NFC) between the bands to share information about the users and Bluetooth communication between the band and the phone in order to present the shared information in an interactive way through smartConnect's mobile application.

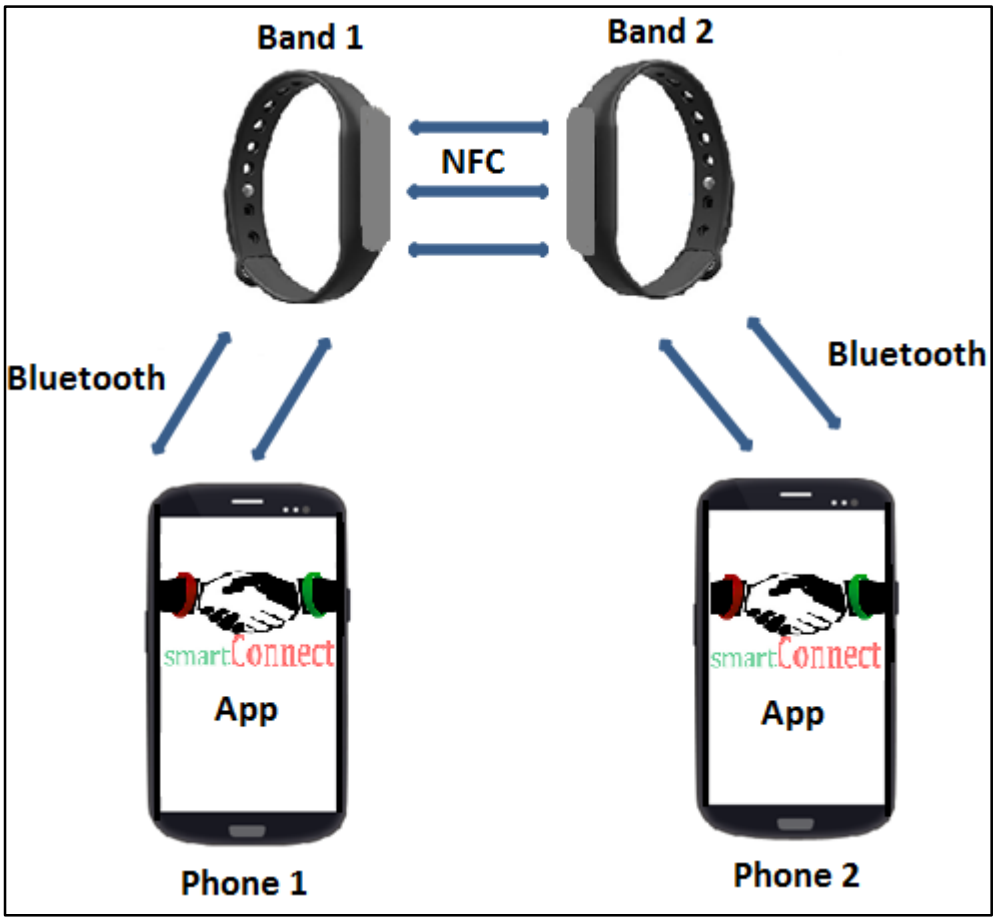



Figure 1: smartBand Concept Art

## 2.2. System Overview

This section provides a high level overview of the prototype version of the smartBand design and the figures shown are the future implementation of the ideal design which is currently implemented using the hardware components labelled on the figure. As shown in Figure 2, there are three main spectations: exchange of user IDs between the bands through NFC, exchange of user ID between the band and the phone via Bluetooth and activation of the NFC reader by a 3-axis accelerometer.

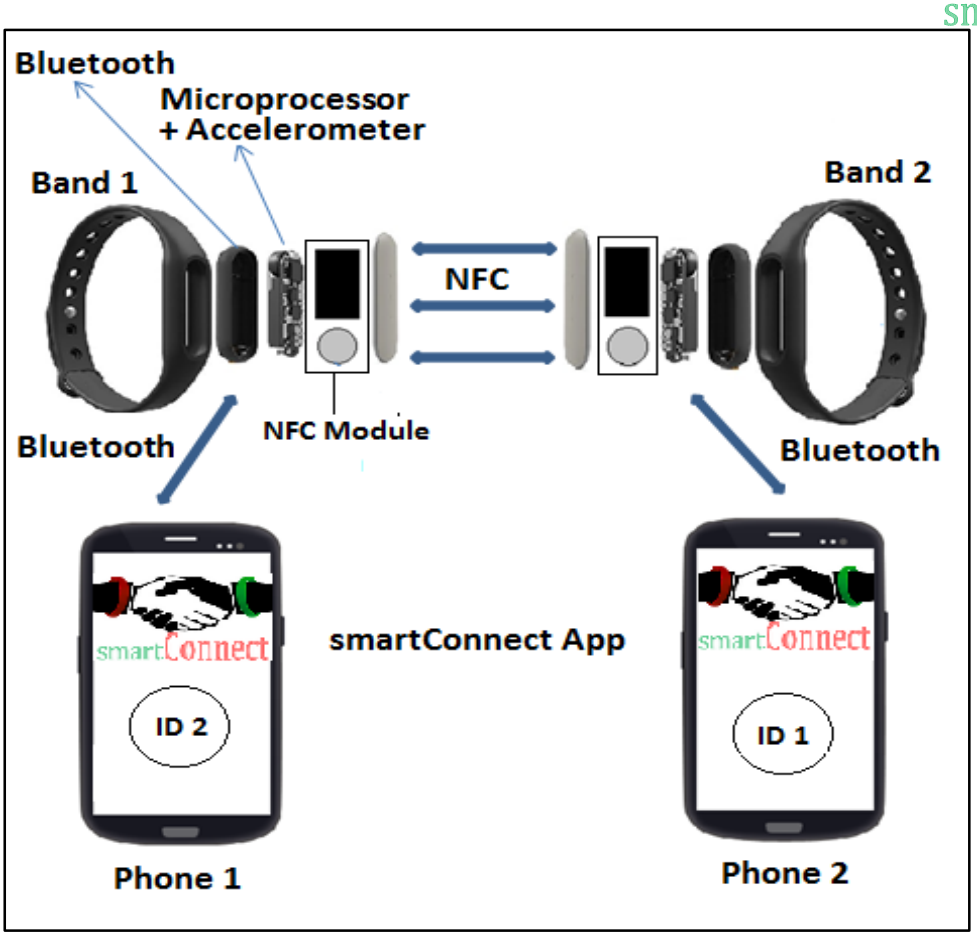



Figure 2: Component interaction in smartBand

Figure 3 below provides the simplified explanation on the interactions between the internal submodules of the device to provide the functional system presented in Figure 2.

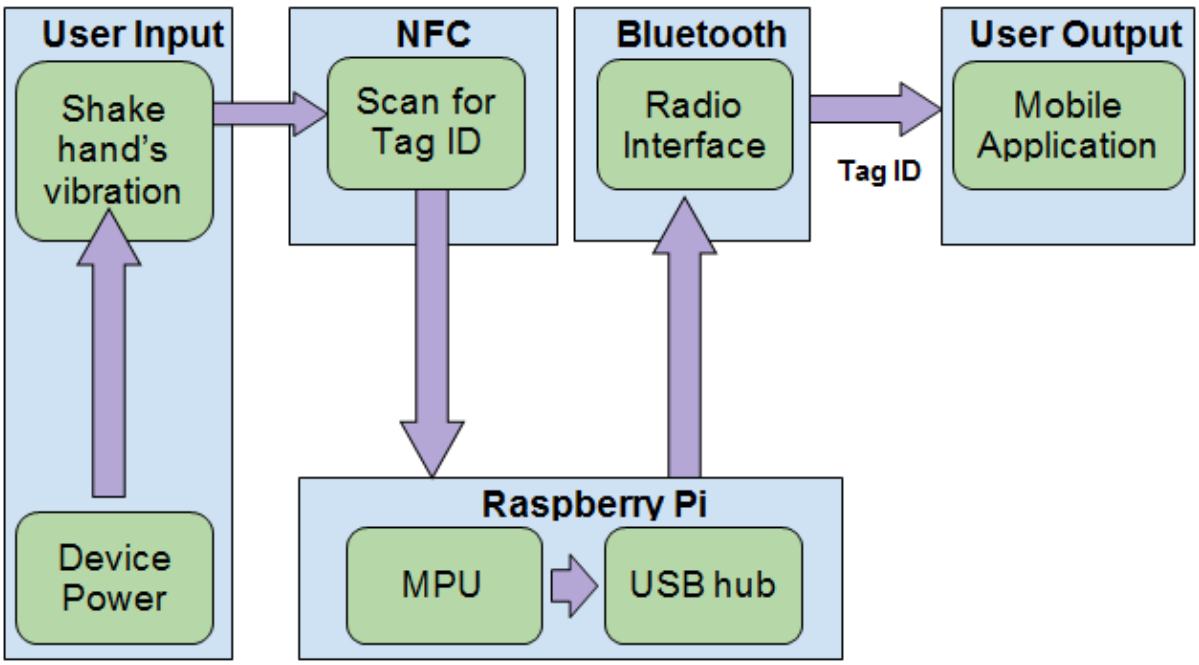



Figure 3: High level system overview - smartBand

The inputs to the system include pressing buttons that turns the device ON, and detection of the vibration from the handshake to start the data transfer between the smartBands. After the handshake activates the NFC reader, it will scan the tag ID of the other band and send it to microprocessor in Raspberry Pi [R-1.3-I, R-1.4-I]. Then the tag ID will be sent to the mobile application running on the user's phone through a Bluetooth signal interaction between the mobile device and Raspberry Pi. The LED on Raspberry pi will provides feedback, showing if the NFC reader is ON or OFF [R-1.12-III] and the details of the implementation are provided in Appendix A. For saving the drivers of the Raspberry Pi, NFC, and Bluetooth modules as well as the python code, we are using 8GB SD in the smartBands [R-1.15-III]. The prototype is powered by 5V and 1A micro USB and will ultimately be optimized into the final product in the shape to a wristband [R-1.8-I, R-1.7-II]. Further design details of each component of smartBand are discussed below in their respective sections.

## 3. Raspberry Pi

For supporting all the sub modules of the system we are using Raspberry Pi 2 Model B as our Microprocessor Unit (MPU). It is the second generation of Raspberry Pi and has 900MHz quad-core ARM Cortex-A7 CPU.

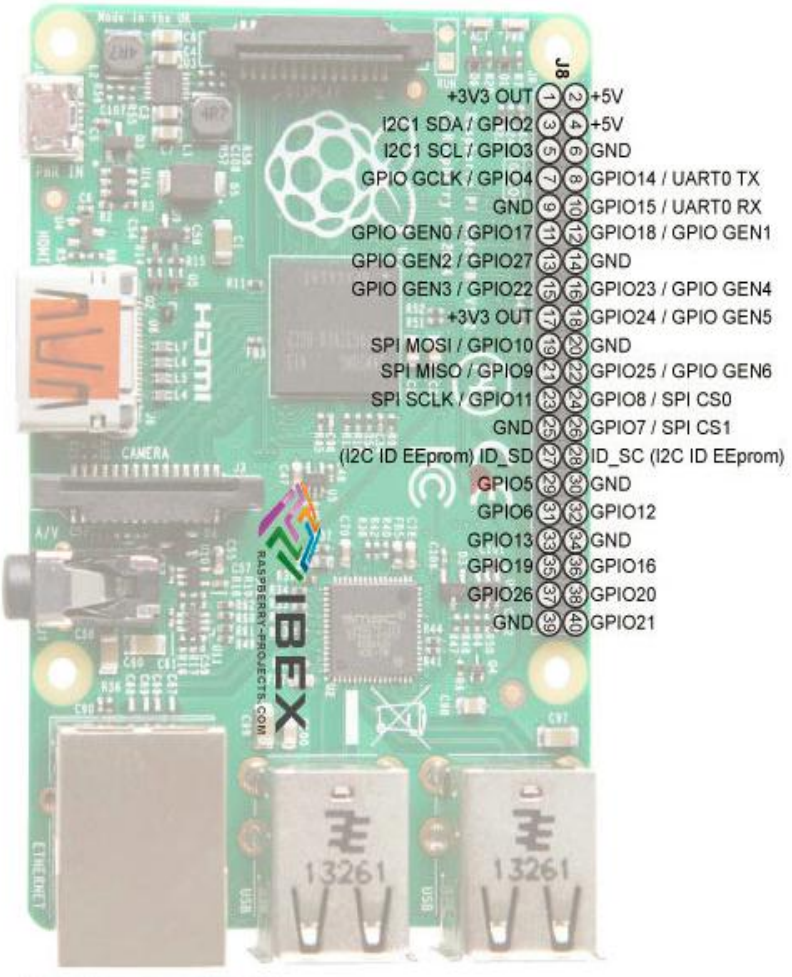



Figure 4: Raspberry Pi 2 Model B with description of 40 GPIO pins [10]

### 3.1. Justification of Use

This model of Raspberry pi MPU was chosen for its lower power consumption as it is powered by only 5V micro-USB [R-1.1-I]. It is also easy to handle, and has high tolerance to temperature ranging from -40°C to 85°C meeting our requirement to work close to room temperature [R-2.5-I]. It has quad-core ARM Cortex-a7 processor enabling it to run the full range if ARM GNU/Linux distributions by running at 900MHz [R-2.2-I]. Also, this model was chosen for the ease of integration into our design with NFC as well as compatibility with the 3-axis accelerometer [R-2.3-I] [10].

### 3.2. Implementation of Raspberry Pi

For communication between two devices, smartband uses NFC and Bluetooth communication. Each NFC module has a tag and reader; a 3-Axis Accelerometer is used to activate the NFC Reader to detect the tag ID from the other band's tag. Once the reader on NFC module reads the tag, it transfers the tag to the programmable MPU which then transfers it onto the mobile application via Bluetooth signal [R-1.3-I, R-1.4-I] (as shown in Figure 3). As shown in Figure 4, the MPU has 40 GPIO pins, I2C, Serial Peripheral Interfaces, 4 USB hub where we are only using 26 pins as discussed the respective sections [6]. The microprocessor (Raspberry Pi) supported by Linux OS, provides the main interaction platform between NFC and Bluetooth module and the connections to these modules are shown in the Figure 5 below [R-2.3-I, R-2.4-I]. For simplicity of the diagram we eliminated the labeling of each of the pins by keeping the spatial arrangement same as in Figure 4. Therefore, in Figure 5 the top right represents Pin 2 (+5V) and the bottom left pin represents Pin 39 (GND) also as labelled the blue line are the connections to the 3-axis Accelerometer and black lines to NFC module [10].

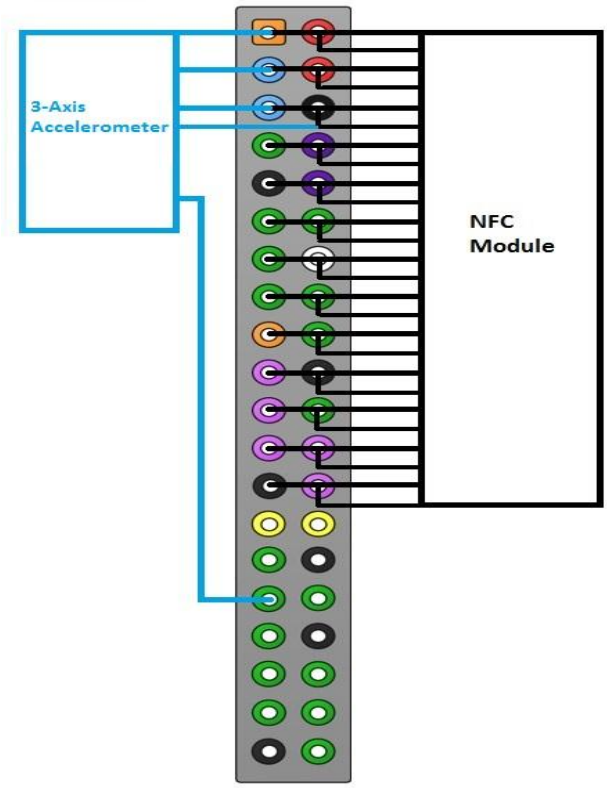



Figure 5: Raspberry Pi connections to NFC Module and 3-Axis Accelerometer

# 4. NFC module (Reader +Tag)

We will be using the NFC model number AN11480 developed by NXP. This device allows peer-to-peer radio communications and transfer data between two devices by touching or putting close to each other [1].


Figure 6: Model AN11480

## 4.1. Justification for use

NFC model number AN11480 was chosen because of its size compatibility with Raspberry Pi making it easier to integrate. This NFC model is highly integrated analog circuitry and also has power less modules [R-3.1.III] [2]. In addition, it was chosen for its lower power consumption that is within 2.5 W to 3.6 W [R-3.2.III] as well as its high tolerance to temperature that is from -30°C to 85°C [R-3.3.III] [3]. We are using NFC for communication between two bands instead of Bluetooth because NFC has lower risk of unwanted interception due to short distance (80mm to 100mm) [R-3.6.III], does not require pairing and has quicker connection and response as comparison to Bluetooth [4].

## 4.2. Implementation of NFC

NFC is a short range Radio Frequency Identification (RFID) wireless communication method where every NFC enabled device contains three components: a tag build with an IC, tag reader and antenna [5]. When an NFC enabled device is brought in the vicinity of another NFC enabled device the IC generate electromagnetic fields which will cause electrons to move through antenna of the other device enabling transmission of data to the reader. Once the chip inside the tag is powered then it will send stored data back and in our case it will be user ID back to read with another radio signal [5]. NFC module will be connected to Raspberry Pi via pin #1 to pin #26 (see Figure 4).
Appendix A contains the code we wrote to program Model AN11480 reader to read NFC tag.

# 5. 3-axis Accelerometer:

The main purpose of accelerometer in smartBand is to detect the vibration caused by shaking hands, which is then used for activating the reader in the NFC module. For the prototype design, smartBand uses the MMA7455 as our 3-Axis Accelerometer, developed by NXP. This device can measure the static acceleration due to gravity or dynamic acceleration due to motion, shock or vibration [6]. The measurements for acceleration are done using the axis system as shown in Figure 7 below [6].

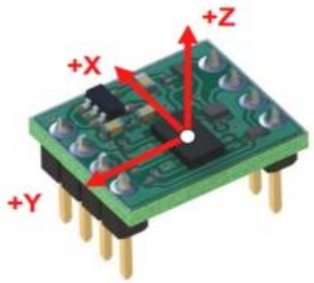


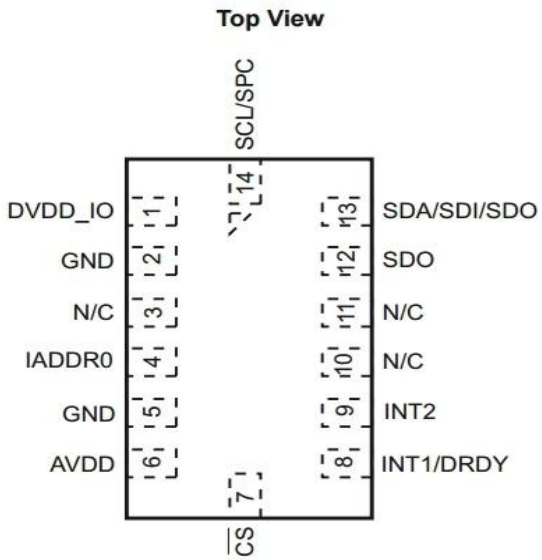

Figure 7: 3-Axis Accelerometer's axis arrangement

### 5.1. Justification for design approach

This is chosen for its small size (3 mm x 5 mm x 1 mm), Low-voltage operation (2.4 – 3.6-volts), very lightweight range of ±2g, ±4g, ±8g for 8-bit mode [R-1.5-I], very low cost [R-1.1-I] and high sensitivity (64 LSB/g @ 2g and @ 8g in 10-bit mode) as well as its shock survival rate (5,000 g shock survival)[7,8]. In addition, it was chosen for the ease of integration into our design.

### 5.2. Implementation in smartBand

The sensor consists of a surface capacitive sensing cell (g-cell) and a signal conditioning ASIC contained in a single package. The ASIC uses switched capacitor techniques to measure the g-cell capacitors and extract the acceleration data from the difference of two capacitors. Also, the ASIC signal filters (by switched capacitor) and provides the digital output that is proportional to acceleration [6]. Appendix B has the Functional Block Diagram for MMA7455 and Figure 8 explains the description of the pins.




| Pin # | Pin Name | Description | Pin Status |
|---|---|---|---|
| 1 | DVDD_IO | Digital Power for I/O pads | Input |
| 2 | GND | Ground | Input |
| 3 | N/C | No internal connection. Leave unconnected or connect to Ground. | Input |
| 4 | IADDR0 | $I^2C$ Address Bit 0 | Input |
| 5 | GND | Ground | Input |
| 6 | AVDD | Analog Power | Input |
| 7 | $\overline{CS}$ | SPI Enable (0), $I^2C$ Enable (1) | Input |
| 8 | INT1/DRDY | Interrupt 1/ Data Ready | Output |
| 9 | INT2 | Interrupt 2 | Output |
| 10 | N/C | No internal connection. Leave unconnected or connect to Ground. | Input |
| 11 | N/C | No internal connection. Leave unconnected or connect to Ground. | Input |
| 12 | SDO | SPI Serial Data Output | Output |
| 13 | SDA/SDI/SDO | $I^2C$ Serial Data (SDA), SPI Serial Data Input (SDI), 3-wire interface Serial Data Output (SDO) | Open Drain/ Input/ Output |
| 14 | SCL/SPC | $I^2C$ Serial Clock (SCL), SPI Serial Clock (SPC) | Input |

Figure 8: MMA7455 Pin Descriptions [6]

Figure 9 shows the wiring of MMA7445 with Raspberry Pi where Accelerometer is connected with Raspberry Pi via $I^2C$.

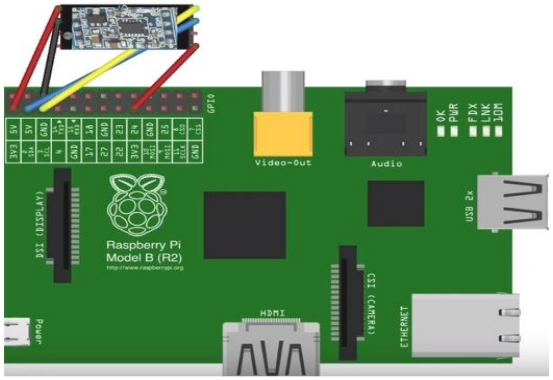



**Figure 9: MMA7455 Accelerometer wiring**

Appendix C and D covers the procedure for configuring Raspberry Pi for I2C.
Furthermore, we will expand the code to sense the vibration of the shaking hands to trigger the NFC reader.

### 5.3. Mathematics Algorithm

Following formula is used to calculate the vibration for stationary device:

$$a_{total} = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

When the device is lying horizontal, $a_x$ and $a_y$ is always zero. However, gravity is always pulling down the z direction, so that $a_z$will be -1. In this case, the calculation of sensing the vibration becomes [9]:

$$a_{total} = \sqrt{a_x^2 + a_y^2 + (a_z + 1)^2}$$

Overall, python code will be used to implement this algorithm.

## 6. Bluetooth Module

For the Bluetooth communication, smartBand uses Bluetooth 4.0 adapter developed by Plugable. Bluetooth provides a faster wireless communication between two or more devices. After a thoughtful comparison between different methods of wireless communication like NFC and Wi-Fi we carefully chose Bluetooth as discussed in this section.

### 6.1. Justification to Use

The Plugable Bluetooth 4.0 adapter support Windows 7 and later, Linux OS as well as is compatible with Raspberry Pi [11]. It is extremely compact USB adapter and is safe while travelling [R-1.23-I]. In addition, it works with Microsoft's new Bluetooth 4.0 Low Energy support on Windows 8® and Windows 10 [11].

### 6.2. Implementation of Bluetooth

In smartBand Bluetooth sends the user ID from the band to the mobile phone in order to obtain user's information [R-2.3-I]. For this wireless communication we considered 3 different technologies: NFC, Wi-Fi and Bluetooth and compared them to decide which one will be the best suited to attain this specific functionality. Table 1 provide the comparison between Bluetooth, NFC and Wi-Fi.

**Table 1: Comparison between Bluetooth, NFC and WIFI [4]**

| Parameter | NFC | Bluetooth | Wi-Fi |
|---|---|---|---|
| Range | Up to 10 cm | up to 30 meters | Up to 30 meters |
| Pairing method | no pairing required | pairing required | no pairing required |
| Speed | 100-400 kb/second | up to 2.1 Mb/second | up to 600 Mb/second |

Since for smartBand we require longer communication range, pairing with mobile phone, and lower power consumption, we decided to use Bluetooth as the transmission medium between smartBand and phone.
Bluetooth communication is employed at two steps in our design; the first occasion being on the band to receive the tag ID of other user from MPU and the second one between band and the phone to send the received tag ID to the users' phone. Appendix E covers further details on the software implementation of the Bluetooth module in the design.

## 7. Battery

In smartBand Battery is used power the Microprocessor, NFC module, Bluetooth module and 3-Axis Accelerometer. We are using 4 Duracell Rechargeable AA batteries to power up of bands. Table 2 provides the operating conditions and functional parameters of the battery.

**Table 2: Specification of Rechargeable AA [12]**

| Properties | Range/Value |
|---|---|
| Nominal Voltage | 1.2V |
| Volume | 8.3 $cm^3$ |
| Operating Temp | -20°C to 50°C |
| Typical Impedance | 25 mΩ @ 1kHz |
| Typical Weight | 28g |

For safety reasons we have to stay lower than or equal to 5V [R-1.13-II]. The reason behind not powering up more than 5V is that we might run into the troubles of burning our components. For instance, if we provide 4xAAA Alkaline batteries where each batter provides 1.5V, resulting in total

of 6V which exceeds the acceptable tolerance range of our Raspberry Pi i.e. 5V and would potentially destroy it. Therefore, we choose to use 4 Duracell Rechargeable AA batteries, providing us 1.2V each and total of 4x1.2V=4.8V which is less maximum tolerance Voltage [12]. The microprocessor requires 5V power and will provide 3.3V and 5 V to NFC module and 3-axis accelerometer. The 3-axis accelerometer operates between 2.4V to 3.6V, NFC module will connect to 5V GPIO pin [12].

# 8. Android application

This section provides details on the mobile application associated with smartBand that displays the user's information in an interactive way.

## 8.1. System User Interface:

### 8.1.1. Profile

The Mock ups below are made in Photo Shop to give a clear picture of how our profile page will be. The profile is subdivided into parts to make the user interface more concise and interactive.

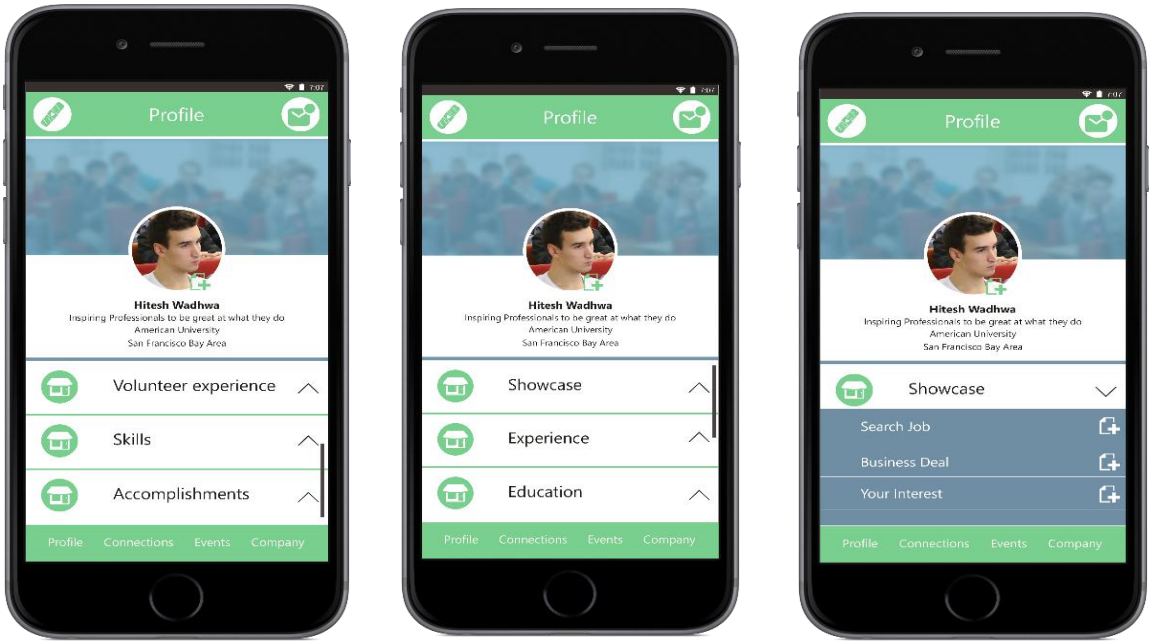



Figure 10: Profile Mock-up

### 8.1.2. Events and connections

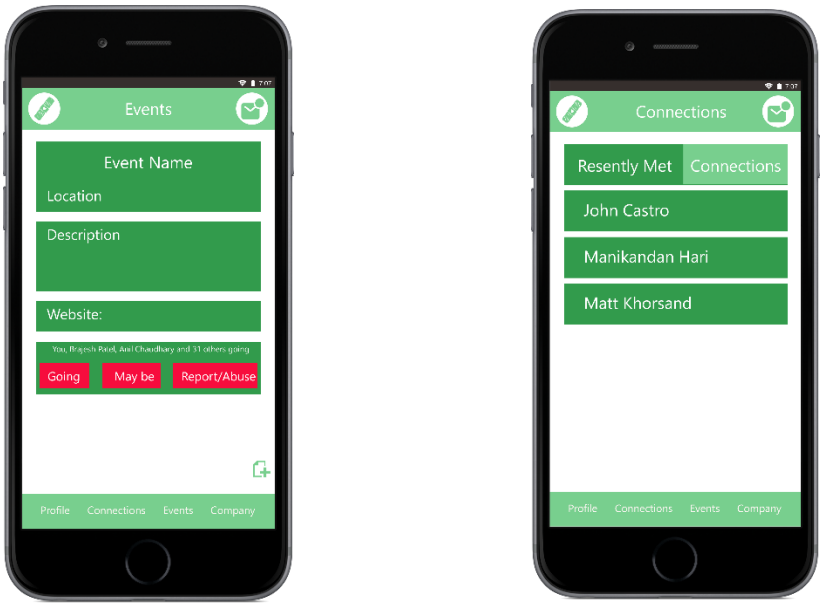



Figure 11: Event and profile Mock-up

## 8.2. Use Case Summary

### 8.2.1. Login

In the beginning of the app the users will be presented to the login page where we have Login option for the existing users, Signup for new users and reset password option in case they forgot his password.

### 8.2.2. Signup

On the Signup page users will be asked to fill in name, e-mail and password. The reason for these limited fill-ups is to make sure that users do not change their mind after looking at a long signup page.

### 8.2.3. Profile setup

After Signup, the users will be taken to the Profile page where they can fill in required information like work experience, projects worked on, Skillsets gained and some personal information. Users will also set up a business card which could be sent to another user. There will be alternate option to export LinkedIn profile (login required) in case users want to skip manual filling.

### 8.2.4. Business

Then after setting up the profile the user can go to the Business Tab to add their Company Page where they can describe their company and projects that they want to showcase in the networking events.

## 8.3. Events

The Events page will have all the posted events where user can browse through and choose options like “going”, “may be going”, and “Report Abuse”. They will also be able to see which of their connections are going to the event. Users are allowed to post their own networking events as well.

### 8.4. Connections

The connections page will have two tabs “Recently Met” and Connections. When a user shakes hand with another user they will show up in “Recently Met” tab. Further, user can decide to either move them to connections or ignore them after viewing their profile.

### 8.5. Use Case

Use case tables for Sign up & Login, Profile Setup, Events section and Connections section is shown in appendix G.

### 8.6. Class Diagram

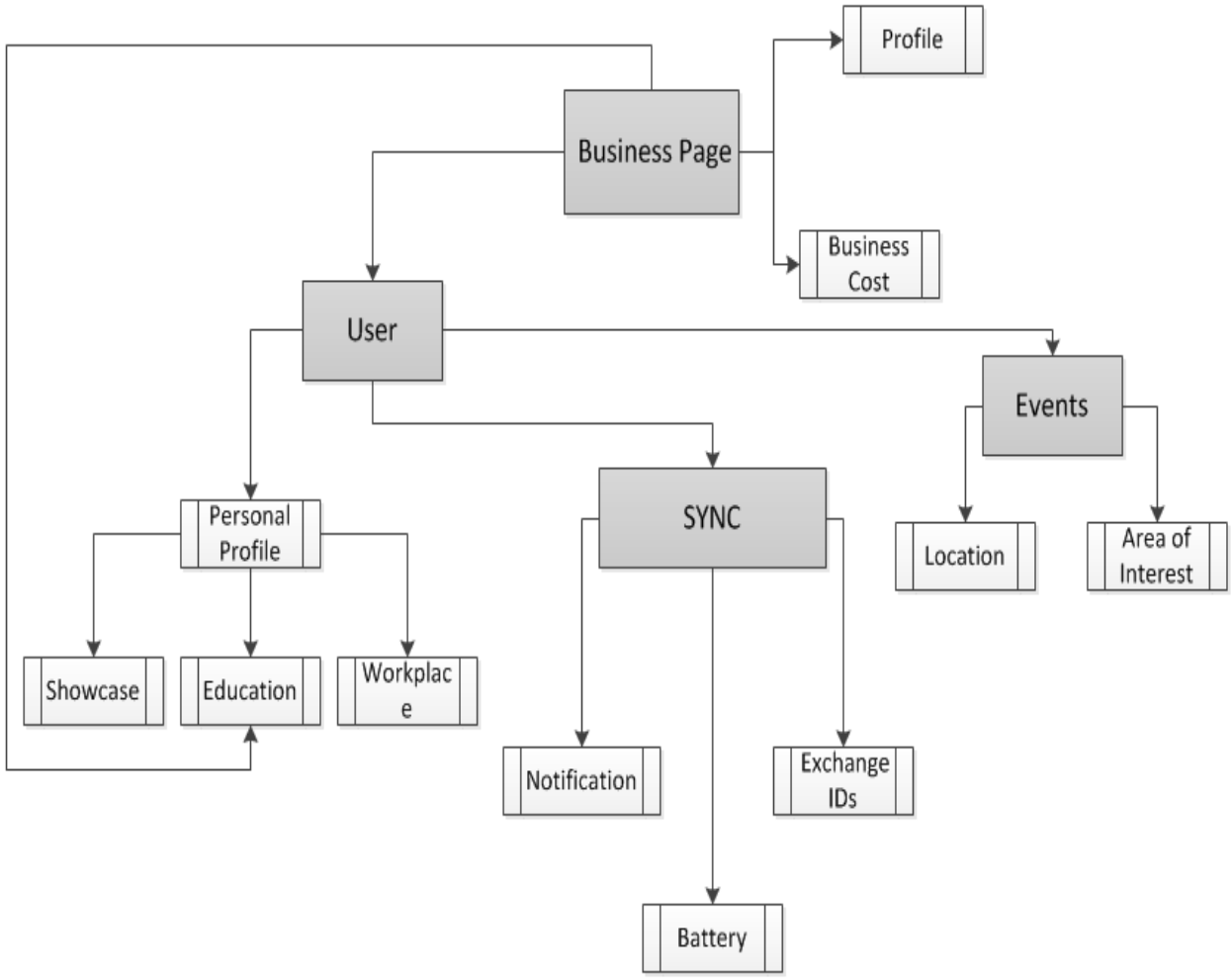



Figure 12: Application class diagram

### 8.7 Data Base Diagram

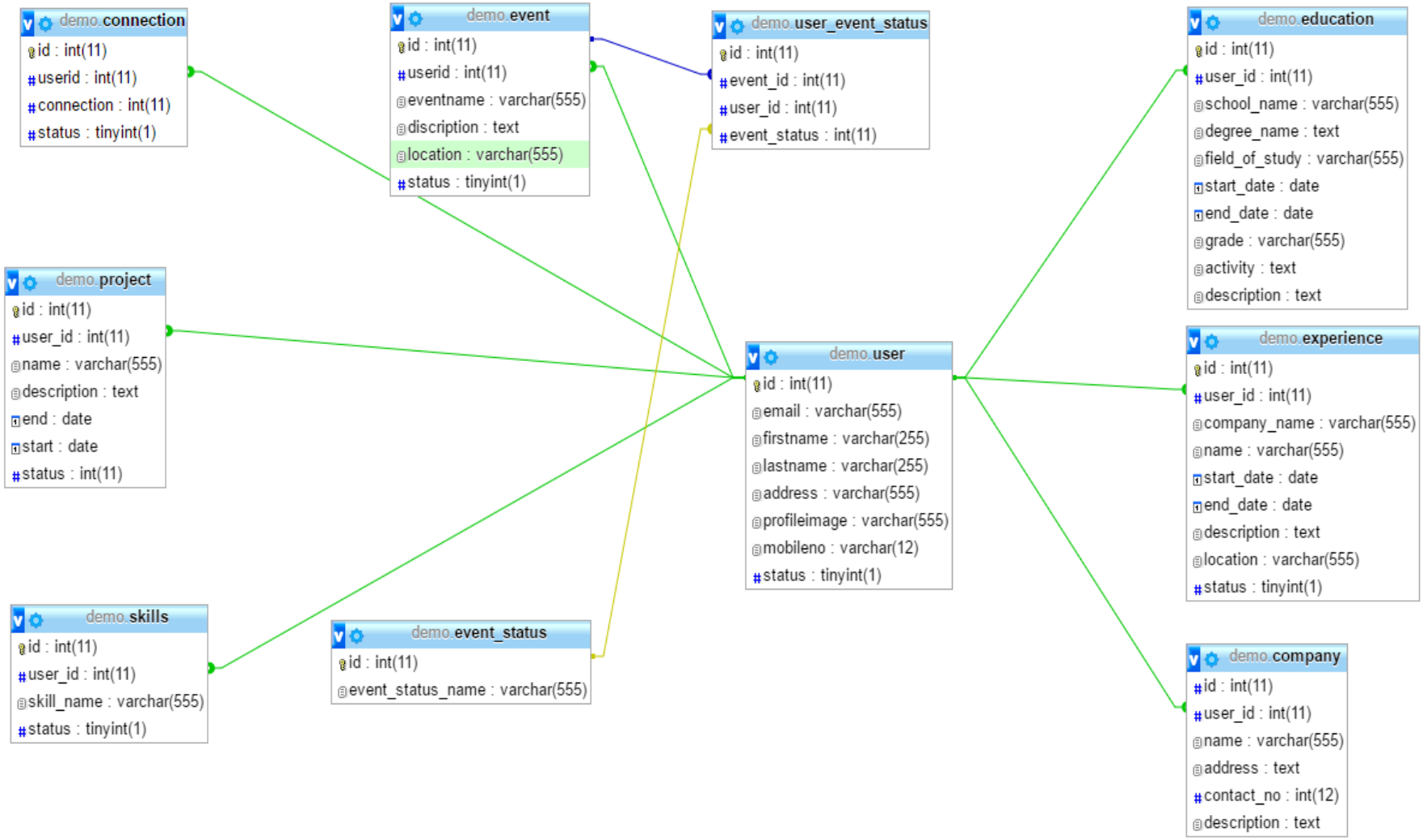



**Figure 13: Database diagram***

## 9. Future Implementation

The current prototype does not possess aesthetically appealing look because we are integrating different hardware modules and programming them together to attain the required functionality. However, in future we will be going for some different components in order to meet the requirements like style, flexibility and wear ability. We might design a CPU that will have same performance as Cortex- A7 and better power efficiency as in the current prototype we are not utilizing all the components use all of their functionalities. For instance, we are only using 26 out of the 40 available PINs on the MPU and using its basic functionality to provide a support between different modules. In order to achieve an optimum utilization of the whole MPU, we will eliminate the unused pins, Ethernet port, audio jack, HDMI port, and camera interface (CSI) by making our own ASIC.

Also we will replace the USB hub by a Lithium Polymer battery that could be recharged through a simple micro-USB plug in order to give the band a slimmer and stylish look. The elimination of the unnecessary elements will help improve the power cost and will improve the look of the band.
We might design our own NFC board by using UCODE 7 chip (this chip has the best performance and have long read distance) [13] to store our user id, and using TRF7960A chip as our read/write IC which requires minimum 1.8V, has dual-input receiver architecture and also supports SPI to communicate with MPU [14]. To improve the communication range we will build our own Inductance Rectangular Planar Spiral Inductor that would help our NFC board to communication within 20-30 cm range [15].

For the prototype we are using Pluggable Bluetooth 4.0 adapter that provides faster connection/ and response time with less power consumption; however, we are limited to certain range. However, the BLE4.2 technologies has less power consumption. Faster data transfer speed, and have better security than the one we using in prototype [16].

We will keep using MMA7455, 3-axis Accelerometer for our feature product, since the sensor is already small enough to fit in the wristband. Also, we will improve our algorithm to detect any kind of shake hand's vibration and prevent those vibrations without interact between two people.

Therefore, all these improvements will help us convert our prototype into the final commercial product by optimizing the factors concerning aesthetics and efficiency.

## 10. Test Plan

### 10.1. Hardware Components (Unit and Integration testing)

**Table 3: Hardware components test**

| Component | Areas/Type of Test | Test description |
|---|---|---|
| **Raspberry Pi microprocessor** | Power through USB port (requires 5V input) | Connect the USB port to different sources of power such as PC, Laptop, and wall sockets and observe that the power indicator LED light turns on |

| | | |
|---|---|---|
| | Pin voltage when turned on | Using a Volt meter to measure the voltage on the pins with assigned voltages from product manual and compare with the values given in the manual |
| **Three-axis Accelerometer** | Three axis values | Moving the Accelerometer should change x, y, z values displayed on Raspberry Pi terminal displayed on PC or laptop. |
| | Device calibration | Connect the Accelerometer to Raspberry Pi that is connected to PC or laptop, and execute the calibration code on Matlab |
| **NFC Reader/Tag** | NFC reader activation | Two group members shake hands while wearing the smartBand. The test is a success when the LED light on Raspberry Pi blinks, indicating the NFC module has been activated by the accelerometer |
| | Input/output test | The led on microprocessor blinks when the NFC reader receives a tag id after a handshake. And to ensure the correct tag ID has been received, the number will be checked on terminal by connecting the Raspberry Pi to a laptop or PC |
| | Distance covered | measure the maximum distance from which the NFC reader can receive the tag ID |
| **SD Card** | Capacity | Use H2testw software to test all memory locations to confirm SD card capacity |
| | Read/Write speed | Use a card reader to plug the card into computer and verify the speed of placing data on the card and opening/running that data. |
| **Bluetooth module** | Connection to Raspberry Pi | Run hfconfig code on Raspberry Pi terminal to see if it detects the Bluetooth module |
| | Pairing with mobile phone | Check to see if the Bluetooth module can be paired with the phone |
| | Distance covered | Measure the maximum distance the Bluetooth module can be detected by the phone |
| **Battery** | The amount of time the batteries will keep the smartBand functioning | Power up the microprocessor using the batteries and keep it on until the device turns off |

## 10.2. Android Application

Table 4: Application test

| Component | Area/Type of Test | Test Description |
|---|---|---|
| **UI** | User acceptance test | Have 10-15 individuals run the application on their android phones and provide feedback on the user interface's convenience and functionality |
| | Performance test | Use dumpsys android tool to verify that the user interactions with the phone are running at consistent 60 frames per second without any dropped or delayed frames |
| **Application Performance** | Manual testing | Developers constantly test any changes they make on local repositories and fix all the bugs found in the code |
| | Asynchronous testing | The application requires SQL database connection and data retrieval. The connection and data download times are tested to make sure the user can access the data with no delays |
| | Integration testing | Check the compatibility of all the application modules with each other and ensure that each module works on its own and is bug-free |

## 10.3. System (Integrated unit testing)

Table 5: System test

| Component(s) | Area/Type of Test | Test Description |
|---|---|---|
| **Android application & Bluetooth module** | Pairing Test | Check to see if the application can detect the Bluetooth ID of the module after pairing the module with the phone |
| **NFC tag, Bluetooth module paired with the phone, and Android application** | Tag ID registration | Use a NFC tag ID to see if Bluetooth module name is overwritten by that ID. Check to see if each ID can be automatically registered for a new account on the application |
| **NFC Reader, Bluetooth module paired with the phone, and Android application** | Connection tag ID input/output test from the Bluetooth module | After a handshake, check to see if the tag ID of another user has been received from the NFC Reader and sent by the Bluetooth module to the application |
| | Tag ID profile retrieval | Send different tag IDs from the Bluetooth module to the application. Check to see if the |

| | | |
|---|---|---|
| | | application can retrieve the correct profile information corresponding to each ID from the database |
| | smartBand to application data transfer timing | Measure the time it takes the Raspberry Pi to receive and send a tag ID to Bluetooth module and also the time it takes for the application to receive the id from Bluetooth module. This determines how long the user has to wait for each handshake. Minimizing this delay is fundamental for this project |

## 11. Conclusion

The design specifications outlined in this document clearly defined the technical requirements and design standards followed towards the implementation of the smart Band's proof of concept model. The product is still under development and final product will be the refined version after performing all the testing mentioned in the test plan. Overall, smartConnect strives to create the optimized prototype for smartBand that is expected to be complete by early April, 2016.

## Reference


[1] Egan, "How to use NFC on your smartphone to do useful things", PC Advisor, 2015. [Online]. Available: http://www.pcadvisor.co.uk/how-to/mobile-phone/what-is-nfc-how-nfc-works-what-it-does-3472879/). [Accessed: 03- Mar- 2016].

[2] "Full NFC Forum Compliant Solution|NXP", Nxp.com, 2016. [Online]. Available: http://www.nxp.com/products/identification-and-security/nfc-and-reader-ics/nfc-frontend-solutions/full-nfc-forum-compliant-solution:PN5120A0ET?fsrch=1&sr=4&pageNum=1. [Accessed: 02- Mar- 2016].

[3] NXP Semiconducor N.V. (2015). “PN512”. [Online]. Available: https://www.nxp.com/ documents/data_ sheet /PN512.pdf. [Accessed: 10- Feb- 2016].

[4] “What is the Difference Between NFC and Bluetooth?”, HubPages. [Online]. Available at: http://hubpages.com/technology/nfc-vs-bluetooth. [Accessed: 02-Mar-2016].

[5] "Transcript | Collin's Lab: RFID | Adafruit Learning System", Learn.adafruit.com, 2016. [Online]. Available: https://learn.adafruit.com/collins-lab-rfid/transcript. [Accessed: 02- Mar- 2016].

[6] “Three Axis Low-g Digital Output Accelerometer”, NXP. 2009, [Online]. Available: https://www.nxp.com/files/sensors/doc/data_sheet/MMA7455L.pdf [Accessed: 02-Mar-2016].

[7] "MMA7455 is one of the cheapest accelerometers with I2C interface", Arduino Projects4u, 2016. [Online]. Available: http://arduino-projects4u.com/mma7455/. [Accessed: 02- Mar- 2016].

[8] "±2g/±4g/±8g, Low g, Digital Ac|NXP", Nxp.com, 2016. [Online]. Available: http://www.nxp.com/products/sensors/accelerometers/3-axis-accelerometers/2g-4g-8g-low-g-digital-accelerometer:MMA745xL. [Accessed: 03- Mar- 2016].

[9] H. axes, "How do I get the total acceleration from 3 axes?", Physics.stackexchange.com, 2016. [Online]. Available: http://physics.stackexchange.com/questions/41653/how-do-i-get-the-total-acceleration-from-3-axes. [Accessed: 3- Mar- 2016].

[10] "Raspberry Pi 2 Model B", Raspberry Pi, 2016. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Accessed: 3- Mar- 2016].

[11] "Plugable USB 2.0 Bluetooth Adapter | Plugable", Plugable.com, 2016. [Online]. Available: http://plugable.com/products/usb-bt4le. [Accessed: 04- Mar- 2016].

[12] "Duracell Rechargeable Cells", Professional.duracell.com, 2016. [Online]. Available: http://professional.duracell.com/en/rechargeable-cells. [Accessed: 03- Mar- 2016].

[13] "TRF7960A | NFC / RFID ICs | NFC / RFID | Description & parametrics", Ti.com, 2016. [Online]. Available: http://www.ti.com/product/TRF7960A. [Accessed: 04- Mar- 2016].

[14] "UCODE|NXP", Nxp.com, 2016. [Online]. Available: http://www.nxp.com/products/identification-and-security/smart-label-and-tag-ics/ucode:MC_50483. [Accessed: 04- Mar- 2016].

[15] Lee, Y. “Antenna Circuit Design for RFID Applications”, Microchip. 2002. [Online]. Available: http://ww1.microchip.com/downloads/en/AppNotes/00710c.pdf [Accessed: 03- Mar- 2016].

[16] "BLE 4.1 vs. BLE 4.2 - New Features and Advantages", Semiconductorstore.com, 2016. [Online]. Available: http://www.semiconductorstore.com/blog/2015/BLE-4-2-vs-BLE-4-1/1548. [Accessed: 03- Mar- 2016].

[17] "karulis/pybluez", GitHub, 2015. [Online]. Available: https://github.com/karulis/pybluez. [Accessed: 1- Mar- 2016].

[18] "Bluetooth programming with Python - PyBluez", People.csail.mit.edu, 2016. [Online]. Available: https://people.csail.mit.edu/albert/bluez-intro/c212.html. [Accessed: 04- Mar- 2016].

[19] svvitale/nxppy", GitHub, 2016. [Online]. Available: https://github.com/svvitale/nxppy. [Accessed: 03- Mar- 2016].

[20] "lauraclay/sensor-projects", GitHub, 2014. [Online]. Available: https://github.com/lauraclay/sensor-projects. [Accessed: 3- Mar- 2016].

[21] "Cortex-A5 Processor - ARM", Arm.com, 2016. [Online]. Available: https://www.arm.com/products/processors/cortex-a/cortex-a5.php. [Accessed: 03- Mar- 2016].

## Appendix A: Code to turn ON LED

We use LED light on the Raspberry Pi to inform the user if NFC reader is powered ON. We decided to blink the LED light while NFC reader is ON, and the following figure shows the sample for turning it ON:

```
from time import sleep
import Rpi.GPIO as GPIO

GPIO.setup(18,GPIO.OUT)
#set up for GPIO#18 (Red LED)

while 1:
    GPIO.output(18,0)
    #Turn on Red LED light
    sleep(1)
    Gpio.output(18,1)
    #Turn off Red LED light
    sleep(1)
```

Figure 14: Sample Code to turn ON LED [18]

## Appendix B: Software Implementation of Bluetooth

We need to install Bluetooth library call PyBluez on Raspberry Pi first. The PyBluez support RFCOMM and L2CAP's Bluetooth socket object, the following is sample code of search mobile phone and send the message on the phone with RFCOMM [17].

```
import bluetooth

target_name = "My Phone"
target_address = None
port = 1

nearby_devices = bluetooth.discover_devices()
#Scan for nearby devices

for bdaddr in nearby_devices:
    if target_name == bluetooth.lookup_name( bdaddr ):
        #connect to each detected devices
        target_address = bdaddr
        break

if target_address is not None:
    print "found target bluetooth device with address ", target_address
else:
    print "could not find target bluetooth device nearby"


sock=bluetooth.BluetoothSocket( bluetooth.RFCOMM )
sock.connect((target_address, port))

sock.send("hello")

sock.close()
```

Figure 15: Sample Code for software implementation of Bluetooth [18]

## Appendix C: Software implementation of NFC [19]

```
import nxppy
import time

mifare = nx.ppy.Mifare()

while True:
        try:
                uid=mifare.select()
                print(uid)

        expect nx.ppy.SelectError:
                #SelectError is raised if no card is in the fiels
                pass

        time.sleep(1)
```

## Appendix D: Software implementation of Accelerometer

```
import smbus
import time
import os
import RPi.GPIO as GPIO

# Define a class called Accel
class Accel():
    myBus=""
    if GPIO.RPI_INFO['P1_REVISION'] == 1:
        myBus=0
    else:
        myBus=1
    b = smbus.SMBus(myBus)
    def setUp(self):
        self.b.write_byte_data(0x1D,0x16,0x55) # Setup the Mode
        self.b.write_byte_data(0x1D,0x10,0) # Calibrate
        self.b.write_byte_data(0x1D,0x11,0) # Calibrate
        self.b.write_byte_data(0x1D,0x12,0) # Calibrate
        self.b.write_byte_data(0x1D,0x13,0) # Calibrate
        self.b.write_byte_data(0x1D,0x14,0) # Calibrate
        self.b.write_byte_data(0x1D,0x15,0) # Calibrate
    def getValueX(self):
        return self.b.read_byte_data(0x1D,0x06)
    def getValueY(self):
        return self.b.read_byte_data(0x1D,0x07)
    def getValueZ(self):
        return self.b.read_byte_data(0x1D,0x08)

MMA7455 = Accel()
MMA7455.setUp()

for a in range(10000):
    x = MMA7455.getValueX()
    y = MMA7455.getValueY()
    z = MMA7455.getValueZ()
    print("X=", x)
    print("Y=", y)
    print("Z=", z)
    time.sleep(0.5)
    os.system("clear")
```

**Figure 16: Sample Code for software implementation of accelerometer [20, 21]**

## Appendix E: Steps to configure our Raspberry Pi for I2C

1. Open Terminal.
2. Type sudo nano /etc/modprobe.d/raspi-blacklist.conf and add "#" before blacklist for I2C. (This will cancel the blacklist for I2C for ou Raspberr Pi).
3. Next, type sudo nano /etc/modules , and then add i2c-dev and i2c-bcm2708.
4. Type sudo apt-get install python-smbus.
5. Type sudo apt-get install i2c-tools.
6. Last step, type sudo reboot the restart the Raspberry Pi.

To check our wiring, we are using command type sudo i2cdetect -y 0 or sudo i2cdetect -y 1. If we get 1d in figure 10, which means that the wiring is successful [5].

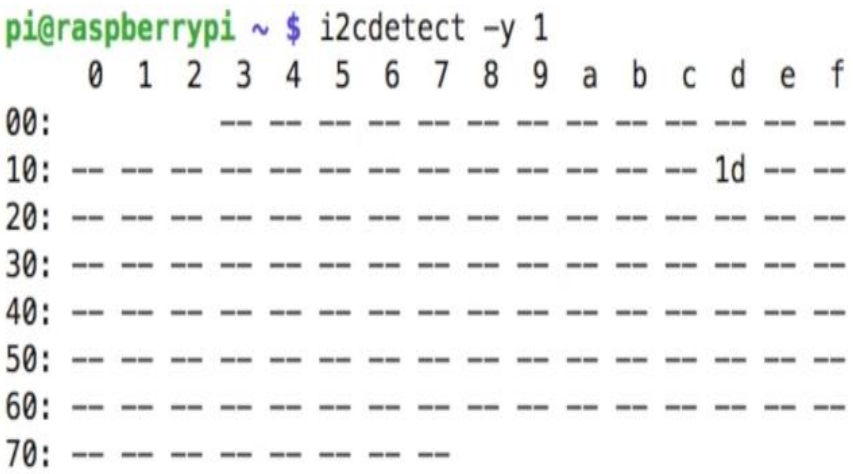



Figure 17: Successful wiring to configure our Raspberry Pi for I2C

## Appendix F: MMA7455 Functional block diagram

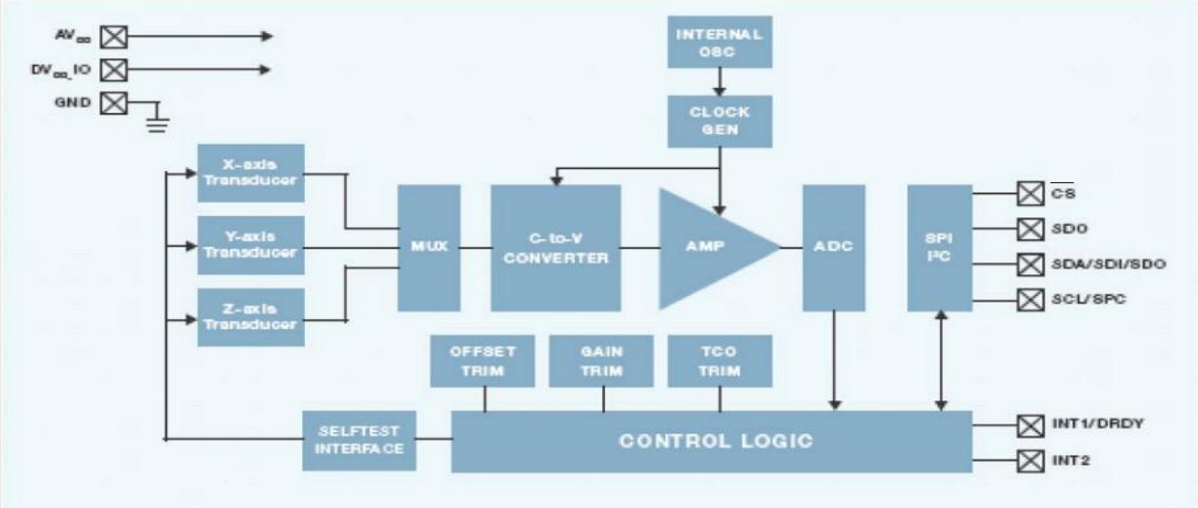



Figure 18: MMA7455 functional block diagram [6]

# Appendix G: Use Case Table

## A. Sign up & Login Interface Design

Table 6: Signup & login design

| | | |
|---|---|---|
| **A** | **Use Case Name & Functional Requirement Number** | **Name:** Application Sign up & Login<br>**Number:** R-4.12 & R-4.13 (Convenient and easy to use interface) |
| **B** | **Participating Actors** | **Primary Actor:** User<br>**Secondary Actor:** Database |
| **C** | **Pre-conditions** | • User should download the smartBand application from app store<br>• The sign up page should be displayed for first use before login<br>• Sql database is used to store login information for auto login |
| **D** | **Main Flow of Events** | • User can input his/her personal information such as: name, last name, email, password, location, etc for sign up<br>• There exists a field that asks the user whether they have an account or not<br>• Choosing it will redirect the user to the login page<br>• The information entered by the user will be saved in SQL database<br>• The user's email address will be his/her username |
| **E** | **Post-Conditions** | • Once the sign-up or login is done, the user will be redirected to profile setup |
| **F** | **Exceptional Flow of Events** | Invalid data entry:<br>o Display message for invalid entry<br>o Point outs the entry with a star<br>o Star appears until the entry is valid<br>Empty entry:<br>o Activity will not be created until all required blanks are filled.<br>o Point out blank entries with the star if "Sign up" Button is pressed before filling them. |

## B. Profile Setup Interface Design

Table 7: Profile page design

| | | |
|---|---|---|
| **A** | **Use Case Name & Functional Requirement Number** | **Name:** Profile Setup<br>**Number:** R-4.3 & R-4.13 (Application must have the option to register the smartBand using the tag id) |
| **B** | **Participating Actors** | **Primary Actor:** User<br>**Secondary Actor:** Database |
| **C** | **Pre-conditions** | The user should be already registered and logged in for profile setup |
| **D** | **Main Flow of Events** | • The user can upload his/her picture<br>• The user has the option to retrieve his/her information from LinkedIn profile<br>• The user has the option to enter his/her projects, work experience, skills, and entrepreneurship plans in this section<br>• The user can enter his personal information such as interests and hobbies in the "About" section<br>• The user can also press the back button to partially save the profile information and complete it later<br>• In case the user owns a company, he/she can create an additional profile for the company to showcase the products and display available job openings |
| **E** | **Post-Conditions** | • Database will be updated with the new or updated entries |

### C. Events Section Interface Design

Table 8: Event page design

| | | |
|---|---|---|
| **A** | **Use Case Name & Functional Requirement Number** | **Name:** Attend/Organize Events<br>**Number:** R-4.8 (Future events will be available for the user to see) |
| **B** | **Participating Actors** | **Primary Actor:** User<br>**Secondary Actor:** Database |
| **C** | **Pre-conditions** | The user should have a valid registered account |
| **D** | **Main Flow of Events** | • A list of upcoming events in the state/province where the user resides in is shown on the events page |

| | | |
|---|---|---|
| | | • The information about date, location, and details of each event is provided with it<br>• The user has the option to choose to attend or ignore an event via the +/- buttons next to it<br>• The user can organize his/her own event via the "organize event" button on top right<br>• If a person from the user's contacts list is attending an event, it will be displayed under that particular event's description |
| **E** | **Post-Conditions** | • Events created by the user will be added to the database<br>• Contacts will be able to see if the user has decided to attend an event in his/her profile page<br>• The event created by the user will be added to the list of events for other users of the smartBand application in the region |
| **F** | **Exceptional Flow of Events** | Invalid data entry for event organization:<br>o Display message for invalid entry such as incorrect date and time<br>o Point outs the entry with a star<br>o Star appears until the entry is valid<br>Empty entry:<br>o Activity will not be created until all required blanks are filled.<br>o Point out blank entries with the star if "Organize" Button is pressed before filling them. |

### D. Connections Section Interface Design

Table 9: Connections page design

| | | |
|---|---|---|
| **A** | **Use Case Name & Functional Requirement Number** | **Name:** Connections & Connection Profile Display<br>**Number:** R-4.7, R-4.8, R-4.9 (The user should be able to save or delete a connection, see connection's future events, and request contact info such as phone number and email address) |
| **B** | **Participating Actors** | **Primary Actor:** User<br>**Secondary Actor:** Database |
| **C** | **Pre-conditions** | • The user should have a valid registered account<br>• The user should have added connections |

| | | |
|---|---|---|
| **D** | **Main Flow of Events** | • The user can view all of the added connections in "Connections" page<br>• The user has the option to delete each connection<br>• The user can view each connection's profile by pressing the connection's name<br>• The connection profile will display:<br>  ○ Profile picture with name and last name<br>  ○ Company owned (if applicable)<br>  ○ Future events to be attended by the contact<br>  ○ Employment history and status<br>  ○ Projects and accomplishments<br>  ○ Skills<br>  ○ Current project and entrepreneurship plans<br>  ○ Personal information (interests and hobbies)<br>• The user can also request email address and phone number by pressing the "Request contact info" button at the bottom of the connection profile page<br>• In case the connection owns a company, the user can view the company page via the link provided below the connection name |
| **E** | **Post-Conditions** | • Requesting contact info will notify the other user, and ask him/her for permission to share contact's phone number and email address with the user |