Qisong Wang
Matriculation number: S1364207
BEng Hons Project Report HSP 2606

Sensory Assistive Device Interpreting Video for Visually-Impaired Users

April 2016

Original Mission Statement

Sensory Assistive Device Interpreting Video for Visually-Impaired Users

Student: Qisong Wang (s1364207)

Supervisor: Dr Philip J.W. Hands

Subject Area: Visual Processing with Haptic Feedback

Project Background:

Although it is often easy to ignore how much of our daily functioning and social interaction is informed by visual social cues such as facial expressions, gestures and body postures, they do play a vital role in effective communications. Missing these subtle visual cues, individuals with a visual impairment would, however, have trouble in receiving the information they convey and making proper responses, which is likely to make them feel isolated and impose negative impacts on developing positive self-esteem, building social bonds and finally the acceptance into society.

Project Overview:

The aim of this project is to design and build a camera and wearable device system for visually-impaired users. Specifically, as shown in Figure 1, camera or camera glasses would first record video of surroundings and stream it to a base-station computer via a USB cable or wireless connectivity (Bluetooth or Wi-Fi). This base-station computer could be a PC, a single board computer like Raspberry Pi or mobile phone for portable, economical and fast-prototyping purposes. Image analysis software running on it would then make decisions about the social situation in real time, and send vibration alert to users via a vibrator (inside a smart watch or haptic glove). This feedback should be intuitive and easily understood by users and should not distract them and their counterparts from normal conversations. The form of feedback may be extended by audio or thermal feedback, if necessary. This system will, therefore,

aid the blind and partially sighted in receiving visual social cues and maintaining social interaction.

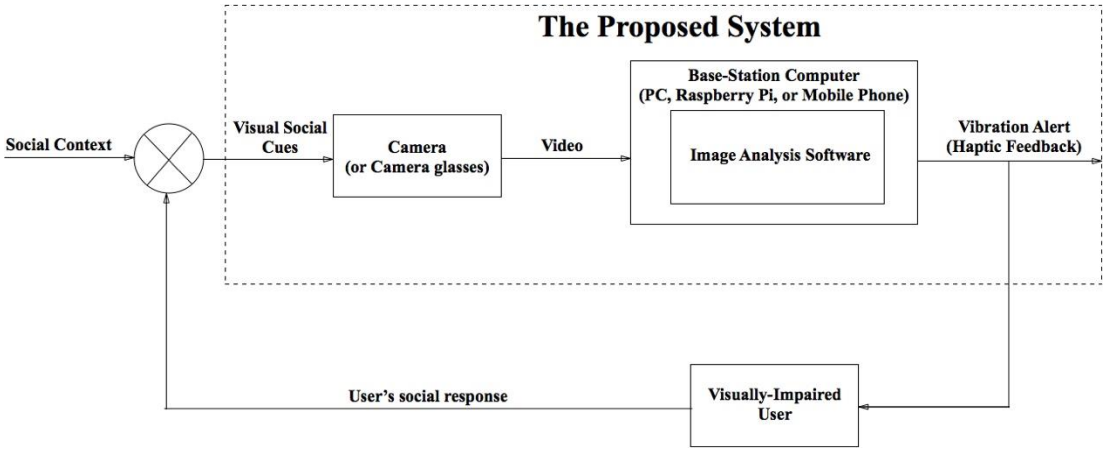


Figure 0: A Block Diagram Illustrating the Proposed System

Project Tasks:

The general strategy of this project is first to prototype the system with software and conventional computing hardware. Later, this will be implemented into a wearable solution, for example, camera glasses and smart watch. The project can be broken down into the following tasks:

Preparatory Tasks:

- Study computer vision and machine learning basics.
- Study state-of-art visual cue recognition algorithms.
- Study and design intuitive haptic feedback patterns.

Main Tasks:

1. Discuss with collaborators from the University of St. Andrews and the University of Stirling.
2. Design, build, program and test a vibrator controller circuit prototype.
3. Design, print and test a PCB for the vibrator controller circuit.
4. Implement basic recognition algorithms in Python (or C/C++) and OpenCV, including:
5. Face detection and recognition
6. Facial expression recognition
7. Body language and posture recognition

8. Gesture recognition
9. Test and improve recognition performance (accuracy, anti-shake, for example), probably with a wearable camera (or camera glasses).
10. Integrate, test and package the whole system.

Scope of Extension:

- Test the system and assess its performance with real visually-impaired or blind-folded-sighted volunteers.
- Machine learning:
- Automatically and continuously improve the decision accuracy with users' feedback.
- Be able to remember and recognise new faces.
- Add other forms of feedback such as sound, thermal pulse to make it more intuitive for users.
- Add proximity (indicating physical and possibly psychological closeness) detection feature, probably by introducing a second sensor.

Background Knowledge:

- Python, C/C++ and Android programming
- Image processing
- Circuit prototyping and PCB design

Resources:

- Dr Juan Ye, University of St Andrews (Computer science, face recognition and machine learning)
- Dr Ross Goutcher, University of Stirling (Psychology)
- SMC FabLab+ (Prototyping, 3D printing, laser cutting and PCB manufacture)
- TLG and TLF (Computing labs)
- TLD and TLE (Electronics labs)
- OpenCV, scikit-learn, scikit-image (Open source Python libraries)
- PyCharm, Android Studio, CodeBlocks, Arduino (Software IDEs)
- TARGET 3001, Cadence Tool, LTspice (PCB drawing and simulating software)

Abstract

A wearable device has been created, especially for aiding visually impaired individuals in understanding the visual cues (facial expressions) in the daily conversation. The device is capable of generating strong and informative alerting feedbacks, which enables the user to understand his or her conversation counterpart's emotional state. A mobile app and relating cloud service are added to this project. They ensure that the device is always able to produce the most intuitive patterns for the user, which will consequently significantly reduce the time needed for the user to get familiar with the device.

Declaration of Originality

I declare that this thesis is my

original work except where stated.

.....

Statement of Achievement

In this project, I have successfully build a wearable device specially designed for the visually impaired. It is able to play various vibration and light patterns to indicate the emotional state of the person the visually impaired communicates with. The design process has covered everything, including the hardware, software and mechanical design of this watch strap. I have learnt low-level hardware communication protocol I2C and SPI. I have also been exposed to wireless communication standards including Wi-Fi and Bluetooth. I also have learnt how to make real stuff using CAD software.

Besides the watch strap itself, I have designed and build a crowdsourcing platform for the device constantly to improve its vibration or light pattern's intuitiveness. This evolves it from a pure thing to an Internet of Thing.

This project has allowed me to use the various knowledge I learnt before, and further horn my skills, particularly in Android programming.

TABLE OF CONTENTS

List of Symbols

K_e: Motor Voltage Constant

ω: Motor Rotating Speed

Glossary

API: Application Programming Interface

ASD: Autism Spectrum Disorder

CPU: Central Processing Unit

eMMC: embedded Multimedia Card

ERM: Eccentric Rotating Mass

GPIO: General-Purpose Input / Output

IC: Integrated Chip

I2C: Inter-Integrated Chip

OLED: Organic Light Emitting Display

PCB: Printed Circuit Board

LRA: Linear Resonant Actuator

SAD: Sensory Assistive Device

SAT: Sensory Assistive Technology

SCL: Serial Clock

SD: Secure Digital

SDA: Serial Data

SPI: Serial Peripheral Interface

SSD: Sensory Substitution Device

TVSS: Tactile-Vision Sensory Substitution

UI: User Interface

WHO: World Health Organisation

Chapter 1 INTRODUCTION

1.1 BACKGROUND

Visual cues such as facial expressions, gestures, and body postures, play a significantly important role in daily communications. Although this fact might often be ignored by people with sound minds and bodies, Mehrabian's findings in 1967 [1] [2] revealed that a surprisingly huge amount of human's feelings or attitude (55%) is commutated through these visual cues (Figure 1-1). Understanding other individuals' emotion and responding properly in a conversation is crucial to developing healthy interpersonal relationships, and have further far-reaching impacts in the personal development. This point is supported by Charles Darwin's unchallenged argument in his famous book, The Expression of Emotions in Man and Animals [3], that humanity is innately capable of recognising facial expressions as a result of evolution imperative for human survival.

Figure 1-1: Percentage of Components in the Communication of Feelings or Attitudes [4]

The consequences of missing visual cues in the communication, therefore, is severe—it is not only likely to make individual's conversation counterparts feel awkward but is also likely to make themselves feel isolated and impose negative impacts on developing positive self-esteem, building social bonds and finally the acceptance into society.

Unfortunately, there are a significant number of people who are struggling to receive these subtle but important visual cues in a world where communication does matter. The majority of them are the visually-impaired, whose population is estimated to be

285 million worldwide (39 million of blind and 246 with low vision), according to World Health Organisation (WHO) [5]. The other group, people suffering from mental diseases such as Autism Spectrum Disorder (ASD) and Schizophrenia, often exhibit deficits in facial expression and emotion recognition as well [6] [7]. They also account for a large population. Reported by Centres for Diseases Control and Prevention (CDC) [8], for example, about 1% of world population is affected by ASD.

As a consequence, there is an urgent need for new technology aids that can help these people recognise emotions in the communication. The device served as a technology aid in this context is usually titled Sensory Substitution Device (SSD), Sensory Assistive Device (SAD), or Sensory Assistive Technology (SAT), which intends to restore individual's capability of perceiving certain defective sensory modality by taking advantage of sensory information from a functioning sensory modality [9].

1.2 Related Works and Their Issues

Numerous vision substitution SADs have been invented for those who are visually impaired, yet only a few have focused on emotional information. The visual information potentially could be substituted by haptic, auditory, light or thermal signals. Relevant works in each category will be discussed regarding their potential in addressing the background problem and some issues of them will be pointed out.

1.2.1 Tactile-vision Substitution

Figure 1-2: Pure-Form-A Virtual Reality System Where User Can Interact with Work of Art through Haptic Rendering

The most popular alternative to vision is probably haptics since historically it is a natural way for the visually impaired person to explore the world with haptic sensation's aids [10]. A substantial amount of research has been carried out, concentrating on Tactile-Vision Sensory Substitution (TVSS). This study is more about projecting an image to tactile space and visual luminance to tactile vibrations [11]. However, this kind of application is often limited by its resolution and is too complex for the aim of this project. Another disadvantage is that such complex system often results in a large device. For instance, the figure left shows a device

with an exoskeletal interface, which is designed for exploration of artworks at museums. This is not suitable for daily use in social context.

Currently, only two projects have focused on informing the visually impaired users the emotion state of the person they are talking to through vibration patterns.

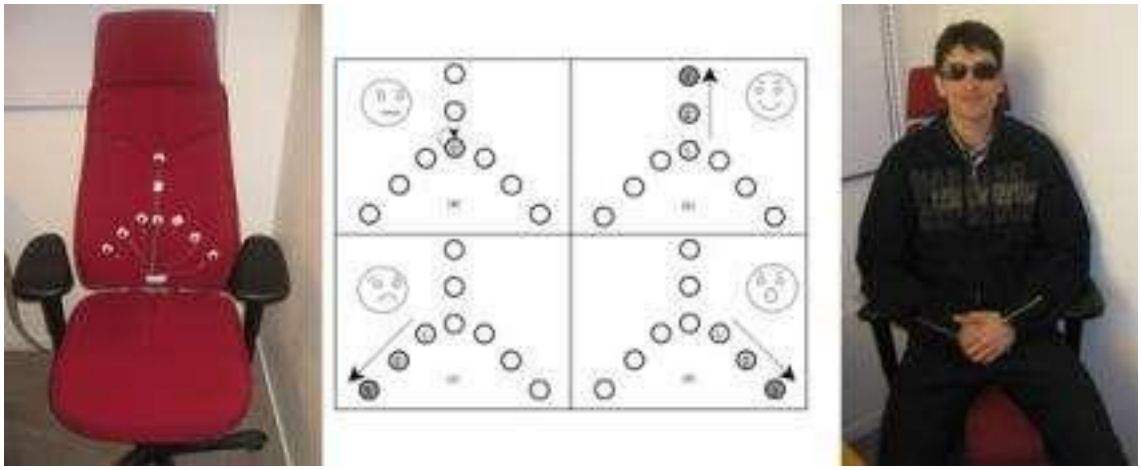


Figure 1-3: A Chair with Multiple Vibration Motors Implementing Brail Code of Emotions [12]

The first one is a chair with vibration motors arranged as Figure 1-3 shows. It can display vibration patterns for four types of emotions, namely 'Neutral', 'Happy', 'Sad' and 'Angry'. The vibration motors in one of three directions will be activated in order if the three non-neutral emotions are detected [12]. The merit of this chair is that the vibration pattern is relatively simple, and the user can quickly learn these patterns by heart and get familiar with them. Nonetheless, this is at the expense of emotional information. In other words, only three of six basic emotions can be conveyed to the user, and it is this loss that makes the intuitiveness of vibration patterns unimportant. Another disadvantage of this chair is clearly its mobility.

The other similar project, VibroGlove [13], its interface is in the form of gloves. According to Figure 1-4 and Figure 1-5, it also locates multiple vibration motors on the hand and it tries to encode vibration patterns according to graphical emotion icons. This encoding method is intuitive for the sighted, but it is doubted whether it is also easy to be memorised by the visually impaired. Furthermore, though it is a wearable device, regrettably, it has limited mobility because its decision signal has to go through a USB cable to play a vibration pattern and it does not have independent batteries, hence is not an entirely wearable device.

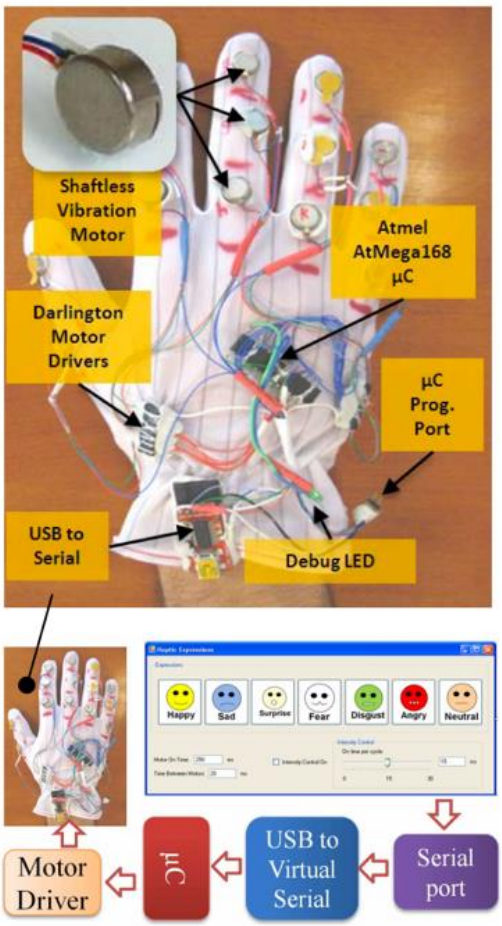


Figure 1-4: System Level Architecture of VibroGlove [13]

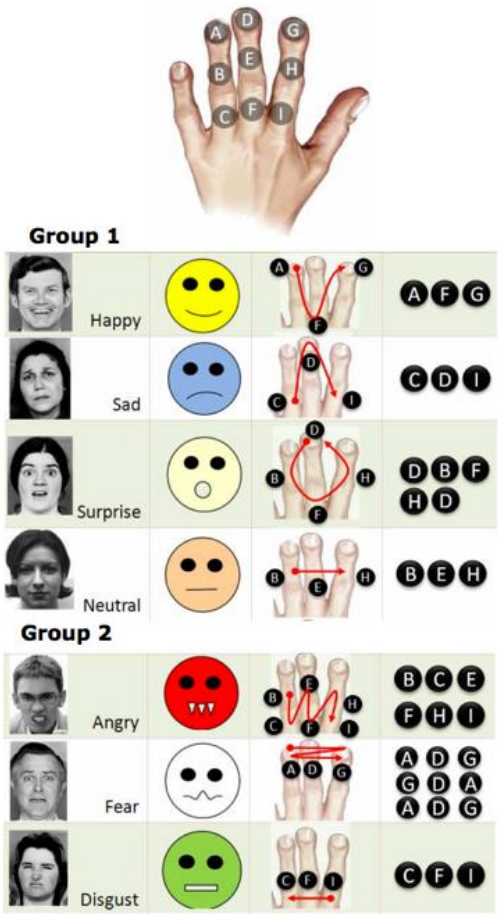


Figure 1-5: VibroGlove's Mapping Methods [13]

A common potential flaw of theses two vibration type SADs is attributed to the use of multiple tactors. This usually implies an increase in complexity. The complexity not only grows in terms of vibration pattern design but also shoots up in terms of user's perception of the overall vibration pattern effect. Consequently, more vibration actuators do not often result in more distinct vibration patterns. In contrast, a previous study investigating "Vibrotactile Target Saliency" has shown that vibration by multiple tactors in a dense area is likely to cause confusion, due to the vibrotactile spatial summation and the occurrence of a vibrotactile version Gestalt Figure [14].

A one more drawback is that there is not too much room left for vibration patterns to be reconfigured in these devices. Individuals, however, can potentially have very different perceptions of the same vibration pattern. Hence, it would be hard for these devices to create highly customisable vibration patterns to cater different visually impaired individual's specific needs.

1.2.2 Auditory-vision Substitution

The other common sensory substitution happens between visual and auditory information. This type of SAD, such as vOICe (Oh I See) project, has been particularly successful in alerting and guiding the visually impaired during walking by utilising depth sensing techniques [9] [15]. However, it may not be useful for the problem mentioned mainly because that an auditory feedback is highly likely to distract the user from a conversation.

1.2.3 Light-vision Substitution

There is no relevant SAT so far using light pattern feedbacks which can consist of colours, texts and flash of colours. However, there is a significant number of psychological research results proving common senses of emotions in colours. Plutchik's emotion wheel [16] (Figure 1-6) showed that emotions can be matched to colours in a colour wheel (a 2D representation of Hue-Saturation-Value (HSV) colour space), where hue is related to the emotion type and saturation and value are related to the intensity of that emotion.

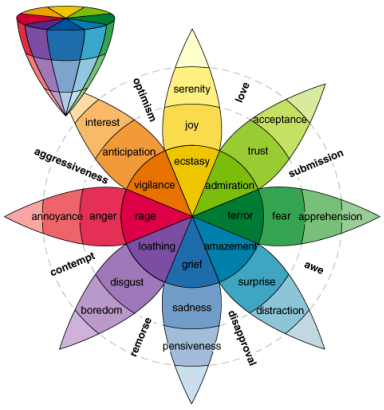


Figure 1-6: The Plutchik Emotion Circumplex 2D (centre) and 3D (top-left corner) Developed in 1980 by Robert Plutchik [16]

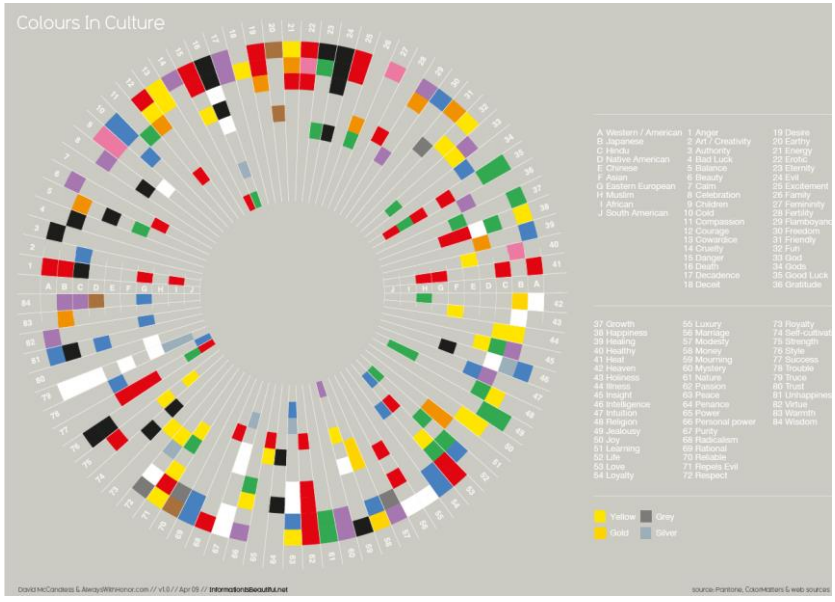


Figure 1-7: Colour In Cultures [17]

Although the colour-emotion association has a relatively high level of intuitiveness, a problem is imposed by its non-universality among different cultures. This can be demonstrated by the figure below. It can be seen that although some colour-emotion associations like mapping between 'Red' and 'Angry" is almost cross-cultural, most

emotions in different cultures are associated with different colours. The design of colour and light patterns thus need to be highly customisable for the users having different cultural and personal backgrounds.

The light alerting pattern is believed to be a worthy compliment to its vibration counterpart, considering the fact mentioned earlier that the blind individuals only occupy a relatively small percentage of the overall potential users.

1.2.4 Thermal-vision Substitution

Thermal information is very intuitive when mapping to emotions, however, the information it can convey is very limited. This is probably the reason why there is no SAD invented based on it so far. Nevertheless, it is believed to be a great compliment to other SAD technologies if adding Peltier cooler/heater module will not result in a far bulky device.

1.2.5 Summary of Issues

It can be seen from the above discussion that, in addition to the size and mobility of the device, the intuitiveness and customisability of alerting patterns are the main concerns. Intuitiveness is strongly desired because unintuitive alerting pattern often will not only result in confusion but also will take a longer time to be learnt. However, it is tough to design "intuitive" patterns by directly applying existing psychological approaches and knowledge. Conversely, understanding how the human brain incorporates external signals into its decision-making processes is often another main aim of a SAD project [18]. It makes no sense to take advantages of knowledge yet be learnt.

1.3 Aim

In the light of this background and motivation, the objective of this project is to design and build a novel SAD and has a focus on intuitiveness and customisability of feedback patterns that this device can generate.

This project is derived from the research [18] conducted by J. Ye et al., its parent project's ultimate aim is to design and build a camera and wearable device system for visually-impaired users. Specifically, as shown in Figure 1-8 and Figure 1-9, camera or camera glasses would first record video of surroundings and stream it to a base-station computer via a USB cable or wireless connectivity (Bluetooth or Wi-Fi). This

base-station computer could be a PC, a single board computer like Raspberry Pi or mobile phone for portable, economical and fast-prototyping purposes. Image analysis software running on it would then make decisions about the social situation in real time, and send vibration alert to users via a vibrator (inside a smart watch or haptic glove). This feedback should be intuitive and easily understood by users and should not distract them and their counterparts from normal conversations. The form of feedback may be extended by audio, light or thermal feedback, if necessary. This system will, therefore, aid the blind and partially sighted in receiving visual social cues and maintaining social interaction.

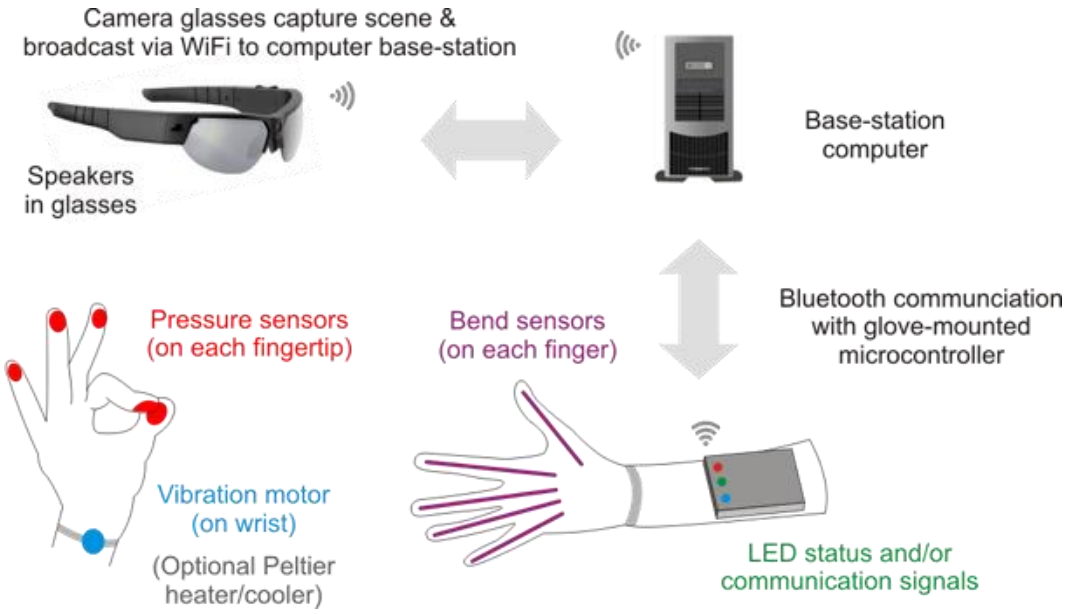


Figure 1-8: Schematic Diagram of the Prototype SAT Hardware. A Vibration motor, Showing An Example Usage of Bend and Pressure Sensors for Gesture Recognition of the User [18]

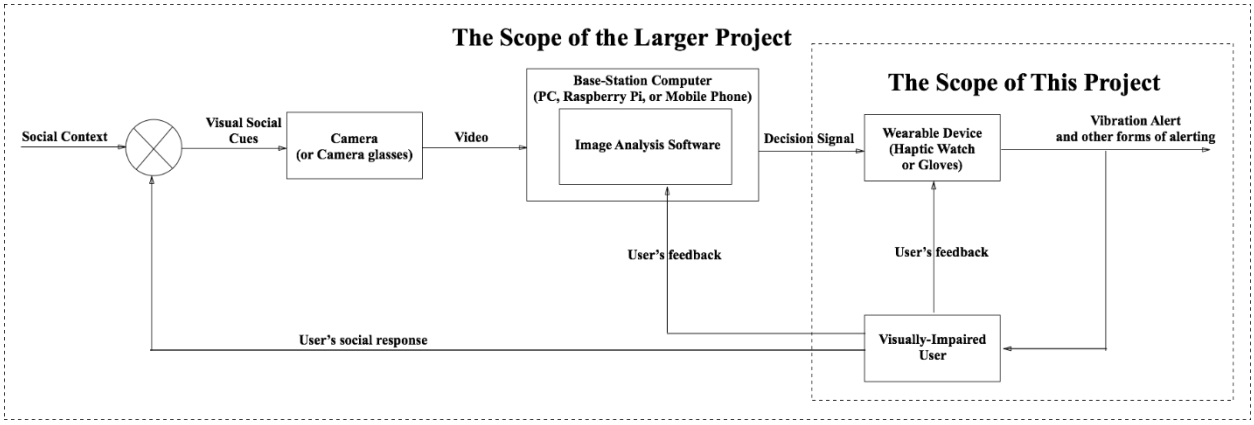


Figure 1-9: Block Diagram Representing the Scope of the Parent Project and This Project

As a sub-project of the bigger interdisciplinary one (illustrated by Figure 1-9), this project has focused more on the engineering part, namely the design and build of the wearable device that can convey basic emotional information (Happiness, Fear, Surprise, Sadness, Disgust and Anger) through intuitive alerting patterns.

The objective of designing intuitive patterns in the original project mission statement has been adjusted to inventing user-friendly tools for designing such patterns.

Beside this, a "crowdsourcing and rating" method is proposed, in response to the issues mentioned in Section 1.2. Its core idea is to develop primitive patterns and gather feedbacks (namely the rating or score of those patterns) from users. Then the pattern design can constantly be improved towards being more intuitive.

This project also aims at supporting users' feedbacks to improve the intuitiveness of the alerting patterns, and the accuracy of the image analysis software.

1.4 Report Organization

The remainder of the report will present more details of the design and development of the device and is organised as follows. Chapter 2 will demonstrate the design of the device itself, in term of hardware, software and mechanical design. Following this, Chapter 3 will briefly discuss the design of a device companion mobile app. The system's integration and the result will be shown in Chapter 5. that, the benefits of this project in the academic, economic and social contexts will be explored. Finally, this report is ended up with a conclusion review the project results and progress.

Chapter 2 Design of Wearable Device

From various considerations mentioned in Section 1.2, the device is required to be simple, small, mobile, and inconspicuous. Hence, the device is decided to be a wearable watch strap, who will provide the users with vibration and light alerting patterns.

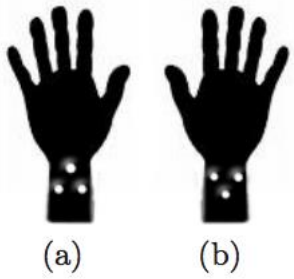


Figure 2-1 : Recommended Tactor Configuration for Wearable Device Worn on the Dorsal (a) and Volar(b) Wrist [19]

Another interesting design decision made is the number of vibration actuators to be used. It is advised by a relevant study [19] that, in order to avoid confusion, human's wrist can at most accommodates three tactors on its dorsal and volar sides respectively, and these tactors need to be spatially localised as shown in the figure below. However, for further reducing the design complexity in both hardware and software, in this project, only one vibration actuator is decided to be localised at the dorsal side of the wrist and is believed to be enough for conveying information.

Overall, the watch design starts by its hardware choices. Upon the completion of hardware design, the software is built, and a shell for enclosing the electronics is designed. What follows will also go through the design process in this order.

2.1 Hardware Design

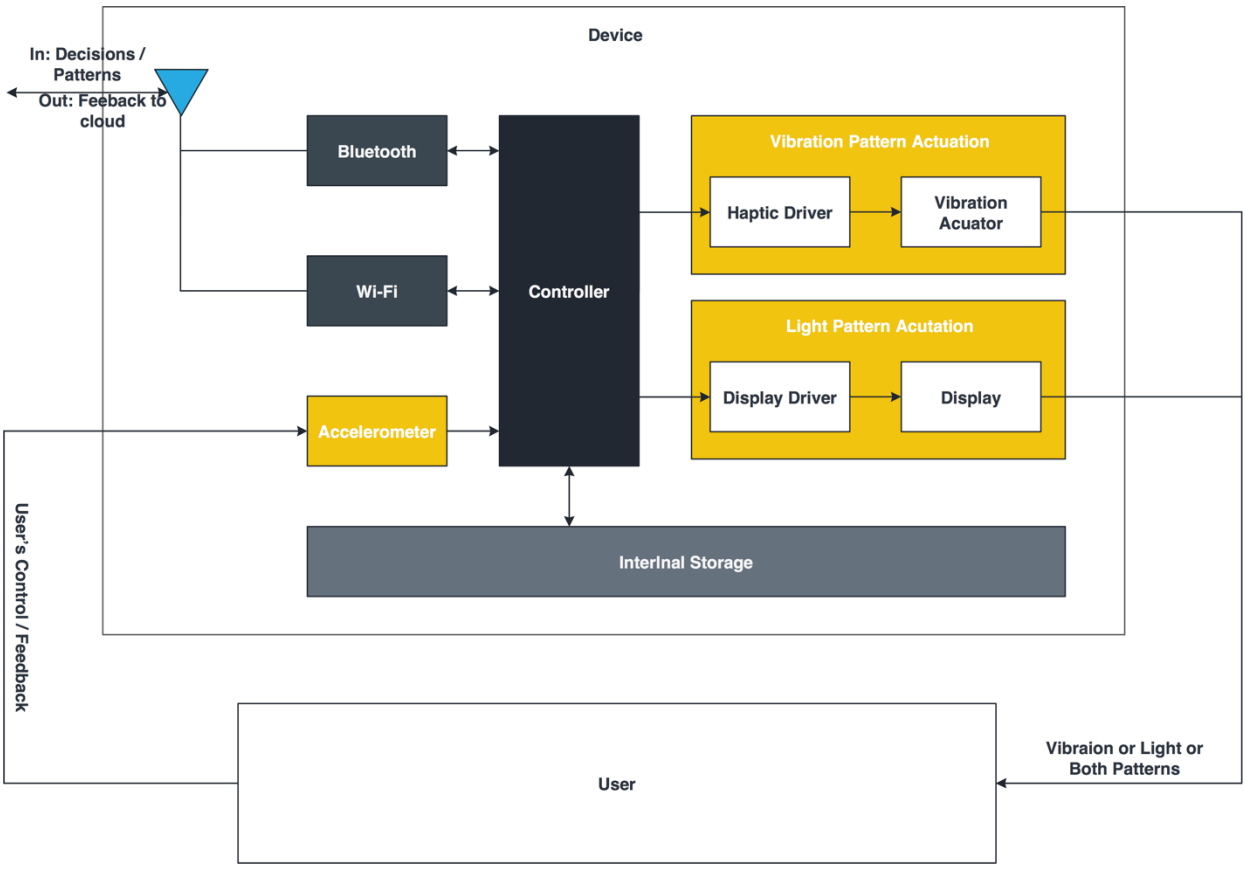


Figure 2-2: Block Diagram of the Device in Interaction with User

The general hardware structure of the design is shown in Figure 2-2. Sitting at the centre position, the controlling unit is responsible for processing and controlling the in and out follow of data. The Bluetooth and Wi-Fi create two portals for the device to communicate with and being controlled by external device or services. The vibration actuator and display are the entities to perform vibration and light patterns. They both need a corresponding driver who is in charge of supplying the device with enough currents and translating low-level controlling signals from a controller to direct electrical signal that the device can understand. These builds up a simple open-loop control system. However, when taking the user into consideration, the system is actually closed-looped with an output-user-accelerometer feedback path, enabling more complex control behaviours. Details of each block are going to be explained.

2.1.1 Controller: Intel Edison Module

Figure 2-3: Intel Edison Compute Module [20]

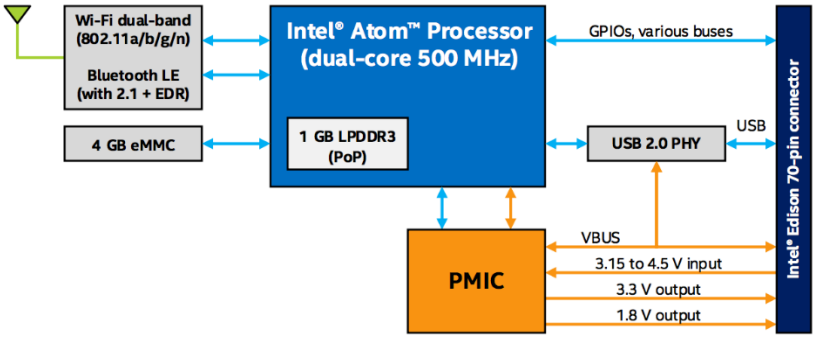


Figure 2-4: Block Diagram of Intel Edison Module [21]

Intel Edison module (Figure 2-3) is chosen as the controlling unit of the wearable device. The block diagram of its hardware structure is shown in Figure 2-4.

Compared with existing controller boards in the market (Table 2-1) [21] [22], it has several unique advantages as far as this project is concerned. Firstly, it features a high-performance and low-power consumption CPU, 2 I2C buses and 1 SPI bus for controlling slave modules. It also has a USB port, a built-in Bluetooth module and a dual-band Wi-Fi module for communication. Secondly, it has a relatively large internal storage, allowing data to be stored locally. Moreover, it supports several programming languages including Arduino, C/C++. Python and JavaScript, offering great

flexibility in the software design. Last but not least, these are all contained within its tiny package, which makes it an ideal choice for prototyping wearable devices.

Table 2-1: Intel Edison Comparison with Arduino Uno and Raspberry Pi [21] [22]

	Arduino Uno	**Raspberry Pi Model B+**	**Intel Edison**
			
Price	$30	$35	$50
Size	7.6 x 1.9 x 6.4 cm	8.5 x 5.6 x 1.7 cm	3.55 x 2.5 x .39 cm
Memory	0.002 MB	512 MB	1 GB
External Interfaces	14 GPIO	1 SD Port, 1 UART, 1 Audio Output, PMW, 1 TWI/I2C, 1 SPI, 40 GPIO	1 SD Port (compatible with SD and eMMC), 2 UART, 2 I2C, 1 SPI, 1 I2S, 12 Additional GPIO
Power Consumption	165 mW (50 mA @ 3.3 V)	3.5 W (700 mA @ 5V)	330 mW (100 mA @ 3.3 V)
Clock Speed	16 MHz	700 MHz	500 MHz, 100 MHz
On-Board Network	None	10/100 BaseT Ethernet Socket	Dual-band (2.4 and 5 GHz) Wi-Fi, Bluetooth 4.0
Multitasking	No	Yes	Yes
Input Voltage	7 to 12 V	5 V	3.3 to 4.5 V
Flash Memory	32 KB	Micro SD Card	4GB eMMC
USB	One, Input Only	Four, Peripherals OK	One, Peripherals OK
Operating System	None	Linux Distributions	Yocoto Linux v2.1
Integrated Development Environment (IDE)	Arduino IDE	Scratch, IDLE, anything with Linux Support	Arduino IDE, Eclipse, Intel XDK

However there two cons of it. The first one is its price, which is relatively high in the same product category. This is allowable in the prototyping stage of the product development. The second disadvantage is its 70-pin connector, which is not very common and requires a base board with a corresponding female connector to break out those pins for use. Xadow Wearable Kit for Intel Edison is an excellent choice for this purpose.

2.1.2 XADOW WEARABLE KIT FOR INTEL EDISON

Figure 2-5: Components Available in the Xadow Wearable Kit for Intel Edison [23]

The Xadow Wearable Kit for Edison (Figure 2-5) is an ideal kit for creating wearable devices with Intel Edison. An Edison expansion board named Xadow-Edison and 8 extremely tiny and useful sensors, actuators, UI modules are included in it.

As shown in Figure 2-6, the core expansion board provides a so-called standard Xadow Interface for connecting other modular PCBs to Intel Edison, and two handy buttons (one for Power/sleep, and the other for Firmware recovery) [24]. The programmer board enables reprogramming Intel Edison and charging the battery. The SD board, however, is not used in this project, since there is enough storage space from Intel Edison itself. These and other modular PCBs can be connected by 12 pin Flat Flexible Cables (FFCs) and share the same interface, which means other modules from Xadow series could also be used. One I2C bus and one SPI bus are broken

out, allowing the use of other standard modules, and is generally enough. This interface is defined in the schematic in Figure 2-7 .

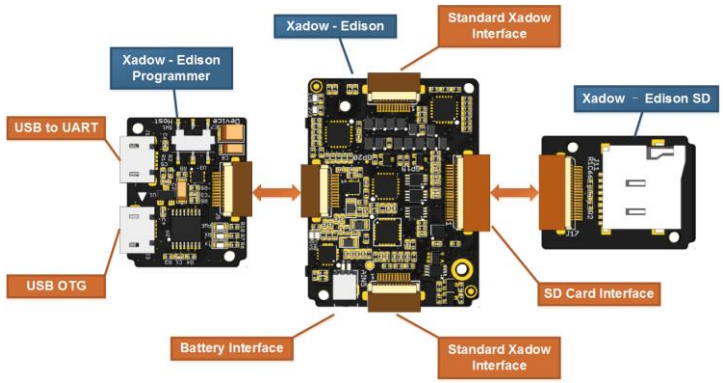


Figure 2-6: Core Components of the Kit [24]

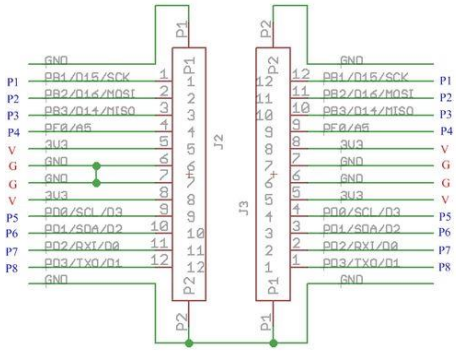


Figure 2-7: Standard Xadow Interface Definition

2.1.3 Vibration Actuator

Nowadays, there are mainly three types of vibration actuators available for providing haptic feedbacks, namely Eccentric Rotating Mass (ERM) motor, Linear Resonant Actuator (LRA) and piezoelectric actuator.

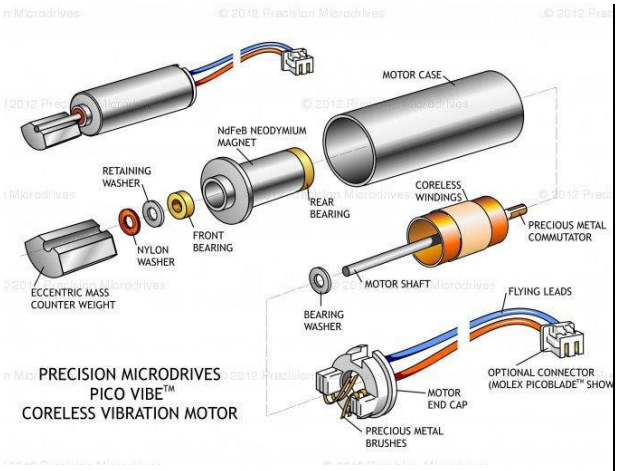


Figure 2-8: Exploded Diagram of an ERM Motor [25]

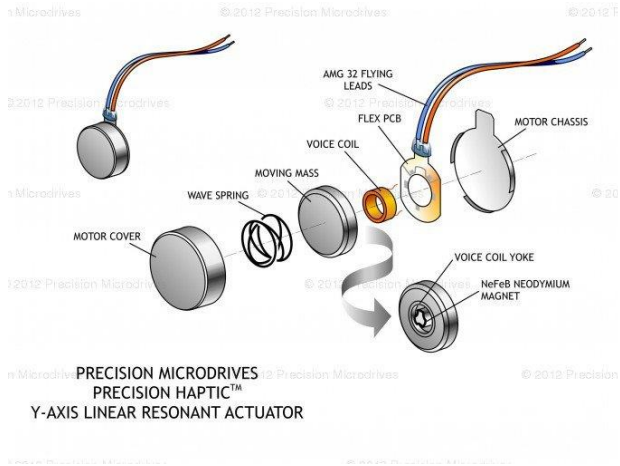


Figure 2-9: Exploded Diagram of an LRA Motor [26]

The ERM motor (Figure 2-8), its working principle is similar to a regular DC motor. It also uses the magnetic field of a direct electrical current to move an object in a circular manner. Especially, the ERM motor moves an eccentric rotating mass (a small weighted object off-centre from the point of rotation). This is where its name comes from. Due to the uneven centripetal force produced by the rotation of the mass, the entire motor will move back and forth to generate a vibration from side to side (lateral vibration). Like most DC motors, an ERM motor will run its current through

windings attached to the shaft of the motor. Since the current will run inside of a magnetic field created by magnets installed inside of the motor, the magnetic field produced by the current will apply a force to the rotating load proportional to the current running through the windings. A simple equivalent circuit model of DC motor can be applied to it as well and an equation relating the intensity of vibration produced by an ERM motor to the supplied voltage at its terminals can be derived.

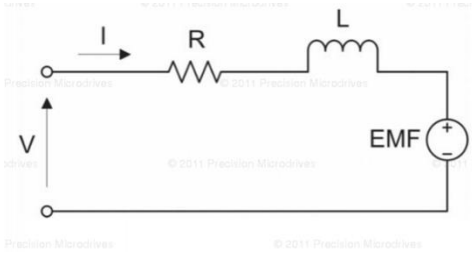


Figure 2-10: Simple Equivalent Circuit Model of ERM Motor [27]

Equation 2.1

$$V = RI + EMF = RI + K_E\omega \text{ [27]}$$

where K_E is the motor voltage constant, and ω is motor speed which is directly proportional to the vibration strength.

It thus can be learnt from Equation 2.1 that relation between the vibration amplitude and the supply voltage will exhibit a relatively linear property, by following Ohm's law. This simple mechanism makes ERM motor available in a wide range of shapes, sizes, and operating characteristics. ERM motors are suitable for applications where a simple and cheap circuit is required and relative low haptic resolution is not an issue [28].

The LRAs (Figure 2-9) have a different mechanism. In this case, vibrations are generated by a movable mass, permanent magnet, voice coil and spring. A magnetic field is produced by the voice coil which is driven by AC signals, and will interact with the permanent magnet, causing it to move and by doing so, compresses or stretches the spring to which it is attached [26]. The amplitude and frequency of the AC drive signals are both important in this case. The vibration amplitude produced by LRA is proportional to AC signal's amplitude. Furthermore, LRA's resonant frequency will also deeply affect its vibration output. To maximise it, LRA must be operated at its resonant frequency [26]. Hence, LRAs are suitable for application which requires independent adjustment of vibration amplitude [28].

The piezoelectric effect refers to an electric charge that accumulates in a solid material as a result of an applied mechanical stress. This circumstance can happen in reverse order. In other words, an electric charge applied to the piezoelectric material

will result in its mechanical change. This reverse process is very useful for generating vibrations. Piezoelectric actuators can be precisely controlled by AC signals, and is even better than LRAs because of they are able to vibrate at a wide range of frequencies and amplitude. The vibration efficiency is better as well since there is no spring. However, currently, they are more suitable for application where ample space is available to house it, such as tablets [28].

As far as the needs of this project are concerned, ERM motor is probably the most suitable candidate, for its easy-to-control feature. A good ERM motor, Precision Microdrive 306-109 [29] is found from Precision Microdrive's 'Haptic Feedback' product catalogue. According to its datasheet [29], This motor features a large vibration amplitude at 3.65 G, which is ideal for vibration alerting and is still able to convey information through subtle vibration changes. This merges the advantages shown in the figure below and is exactly matching the requirements of vibrations this project wants to produce.

Key Technical Specifications

Body Diameter	6 mm	**Body Length**	12.2 mm	**Counterweight Radius**	2.9 mm
Counterweight Length	4.5 mm	**Rated Operating Voltage**	3 V	**Rated Vibration Speed**	12,800 rpm
Typical Rated Operating Current	75 mA	**Typical Normalised Amplitude**	3.65 G		

Typical Performance Characteristics

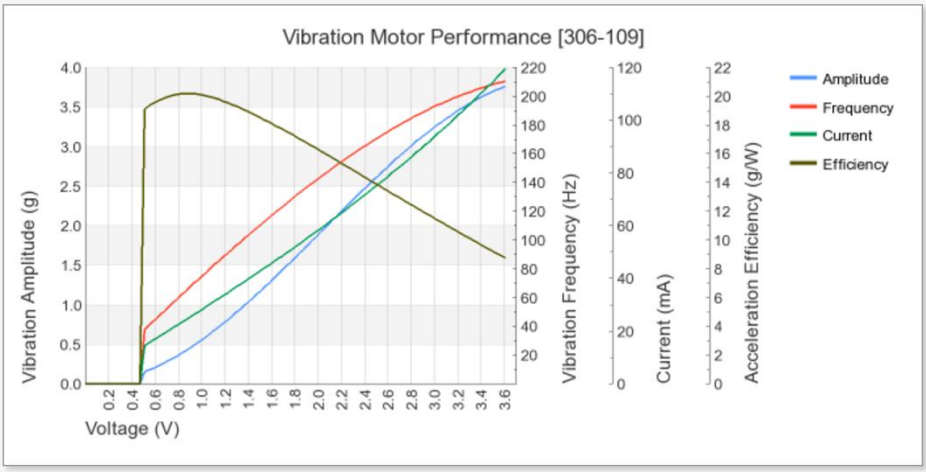


Figure 2-11: Specifications and Characteristics of the Chosen Vibration Motor [29]

2.1.4 HAPTIC DRIVER

2.1.4.1 HAPTIC DRIVER ALTERNATIVES

A haptic driver is needed to drive a vibration actuator because the later requires relatively high voltages and currents to drive, which cannot be supplied from the microprocessor [30]. This can be achieved by simple transistor switch circuit shown below. Advanced discrete MOSFET circuit topology named as "H-Bridge" motor driver, who features active braking for shortening motor rise and stop time is also available (Figure 2-13).

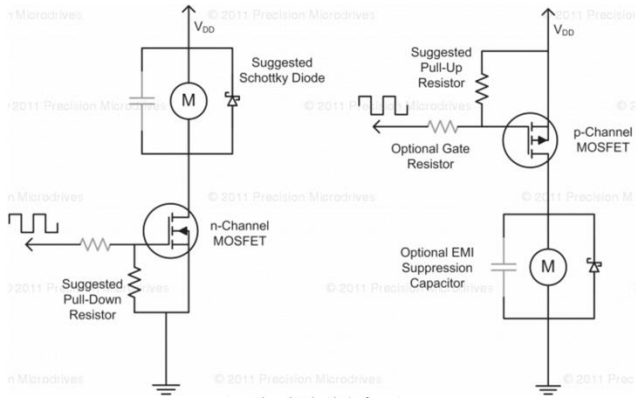


Figure 2-12: MOSFET Driver Circuit in Low-Side and High-Side Configurations [31]

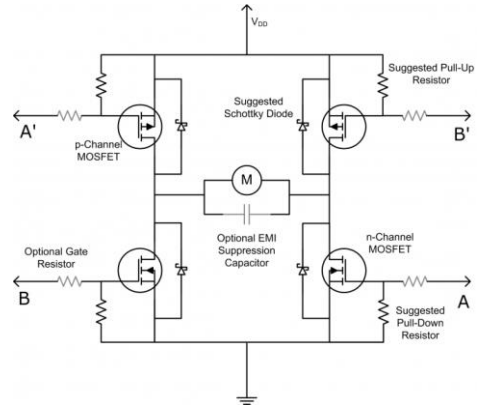


Figure 2-13: H-Bridge Motor Driver Circuit [32]

However, discrete power MOSFET is usually in the TO-220 package, which is far large to be used in building wearable electronics. Fortunately, a variety of integrated motor driver integrated chips (ICs) with much more enhanced vibration controls has been fabricated, including Texas Instruments DRV2605, ON Semiconductor NCP5426 and Maxim MAX1749 [30]. The former and its handy Adafruit breakout PCB is used since there are several pieces left from a previous project.

2.1.4.2 ADAFRUIT DRV2605L PIN-OUTS

The Adafruit DRV2605L haptic driver board (Figure 2-14 and Figure 2-15) breaks out several pins of the surface mounted (SMT) driver IC that is necessary for the microcontroller to interface with it. This include:

- Power Pins (Vin and GND): the motor driver/controller on the breakout requires 3-5V power.

- I2C Pins (SCL and SDA): they are I2C clock and data pin respectively, also requiring the use of 3-5V logic.
- Other (IN/TRIG): A special GPIO that could to 'trigger' the effects to go rather than sending an I2C command. Since it is not useful in this project's situation, it is not used.

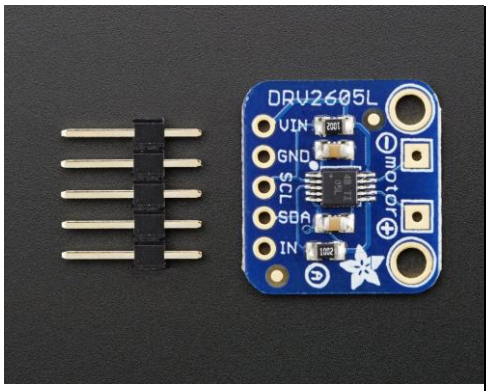


Figure 2-14: Adafruit DRV2605L Breakout Board [33]

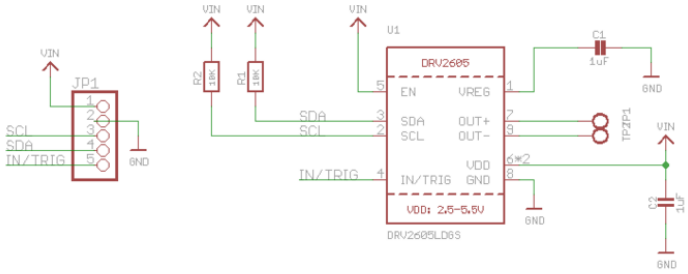


Figure 2-15: Adafruit DRV2605L Schematics [33]

These pins are wired to Xadow–Breakout Board, one tiny PCB from Xadow Wearable Kit and dedicated for the purpose of connecting non-Xadow-standard modules.

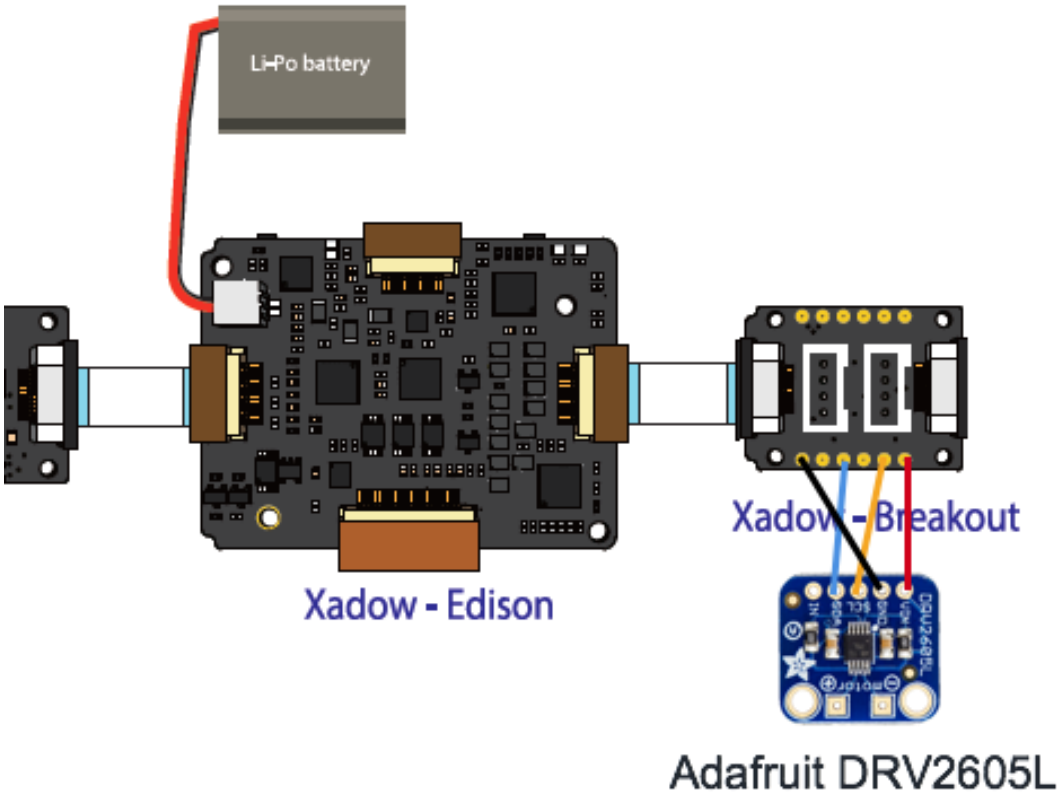


Figure 2-16: Wiring of Adafruit DRV2605L

2.1.4.3 I2C PROTOCOL

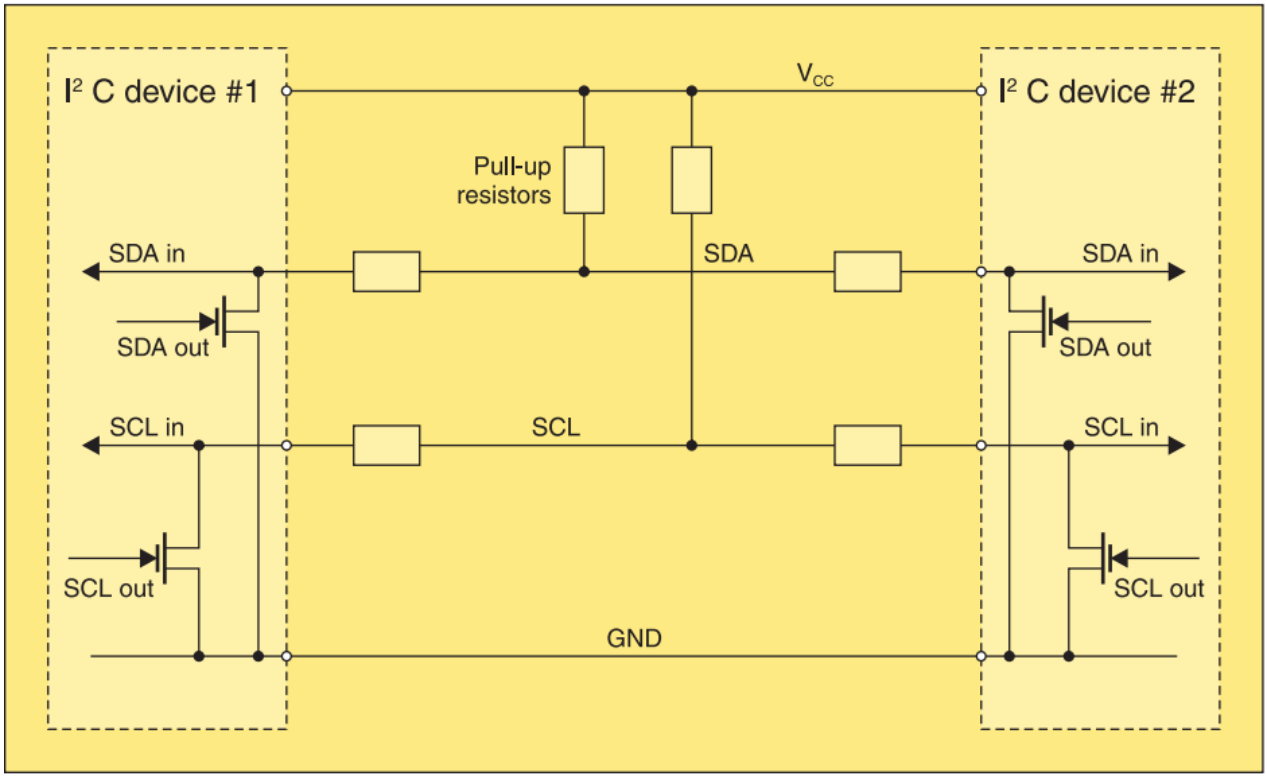


Figure 2-17: Devices Connected Via I2C Buses [34]

I2C protocol (refer to Figure 2-17), which stands for "Inter-Integrated Circuit", is a widely used in the low-end communication between two ICs. It uses 2 signal lines which are called 'serial data' (SDA) and 'serial clock' (SCL) respectively. This protocol allows a master controller to control multiple slaves. Virtually at most 128 slave modules could communicate with the master via these 2 signal line in the light of their unique 7-bits slave addresses.

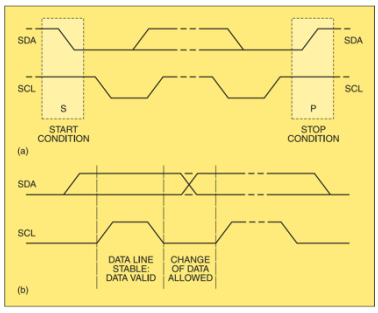


Figure 2-18: I2C Timing Diagram [34]

START	Slave address	Rd/nWr	ACK	Data	ACK	Data	ACK	STOP
1 bit	7 bits	1 bit	1 bit	8 bits	1 bit	8 bits	1 bit	1 bit

Example 1: writing 2 byte to a slave. The data put on the bus by the master are shaded.

START	Slave address	0	0	Data	0	Data	0	STOP
1 bit	7 bits	1 bit	1 bit	8 bits	1 bit	8 bits	1 bit	1 bit

Example 2: reading 2 bytes from a slave. The data put on the bus by the master are shaded.

START	Slave address	1	0	Data	0	Data	1	STOP
1 bit	7 bits	1 bit	1 bit	8 bits	1 bit	8 bits	1 bit	1 bit

Figure 2-19: I2C Byte Data Format [34]

The communication rule and data format are defined by Figure 2-18 and Figure 2-19. The communication is started by the master when it first brings the data line low. This start bit will bring the attentions of all ICs on the bus.

A 7-bit address along with an extra bit indicating read or write operation follows. Having received the address, all IC's will compare it with their own identity. If not match, those ICs will be freed once the stop bit arrives. On the other hand, an acknowledgement bit will be sent from whose identity matching the issued address.

Once the acknowledgement is received, the master will send data to the device. Here data might be a command or a value from the perspective of the slave device. The actions the slave will take really depend on how it is designed and specified in its datasheet. After all data transmission is done, the master will issue a stop bit to end the transmission.

When a master wants to receive data from a slave, it proceeds the same way but sets the RD/nWR bit at a logical one. Once the slave has acknowledged the address, it starts sending the requested data, byte by byte. After each data byte, it is up to the master to acknowledge the received data.

The reading and writing via I2C bus can be easily implemented by taking advantages of Arduino's Wire Library [35]. Taking DRV2605's library [36] for example, they will always start from selecting the target device through "Wire.beginTransmission(DEVICE_ADDRESS)" function, then write/read a byte of data to/from a target device's register.

```
uint8_t Adafruit_DRV2605::readRegister8(uint8_t reg) {
  uint8_t x ;
   // use i2c
    Wire.beginTransmission(DRV2605_ADDR);
    Wire.write((byte)reg);
    Wire.endTransmission();
    Wire.requestFrom((byte)DRV2605_ADDR, (byte)1);
    x = Wire.read();

  //  Serial.print("$"); Serial.print(reg, HEX);
  //  Serial.print(": 0x"); Serial.println(x, HEX);

  return x;
}

void Adafruit_DRV2605::writeRegister8(uint8_t reg, uint8_t val) {
   // use i2c
    Wire.beginTransmission(DRV2605_ADDR);
    Wire.write((byte)reg);
    Wire.write((byte)val);
    Wire.endTransmission();
}
```

2.1.4.4 DRV2605L's Features and APIs

DRV2605L provides two particularly useful features and APIs for creating vibration patterns.

One is so-called Digital Playback Engine. It contains a built-in library of pre-configured haptic waveforms which accounts for 123 in total (listed in Section A.1) [37].

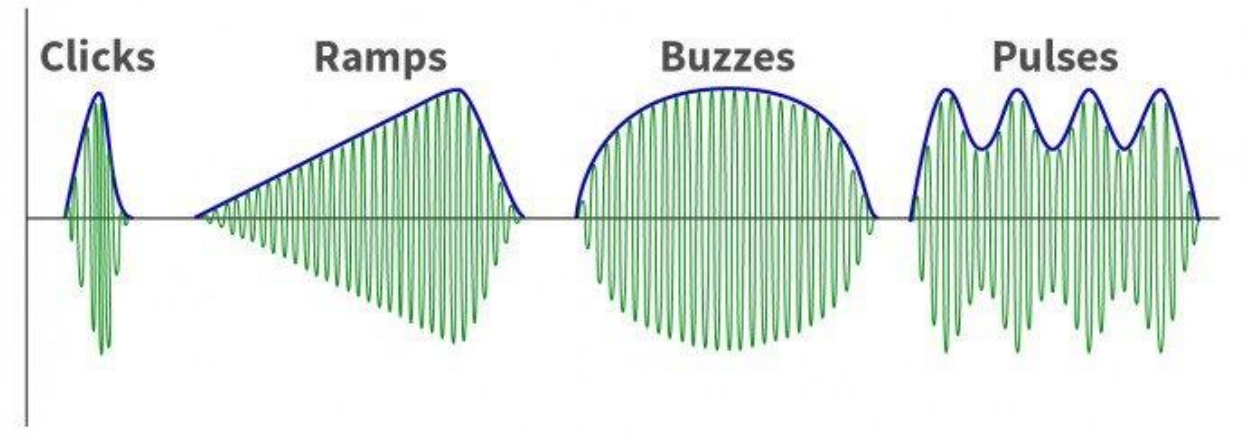


Figure 2-20: Haptic Effects Built upon PWM Signals [38]

These pre-configured library effects generally fall into the above four categories (Figure 2-20) and are actually made up by using Pulse Width Modulation (PWM) [38]. However, with Digital Playback Engine, the user does not need to design patterns from scratch like this, and can design advanced vibration patterns easily based on these simple patterns. These library patterns can be stringed together to form a more complex vibration pattern by using the called "Waveform Sequencer" which is illustrated in Figure 2-21.

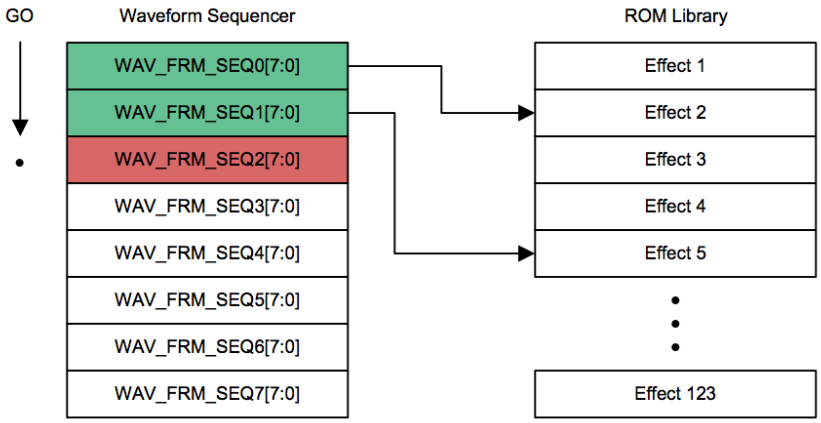


Figure 2-21: Waveform Sequencer Programming [37]

Following is an Arduino program example of it from Adafruit DRV2605 Arduino Library [36]:

```
// I2C trigger by sending 'go' command
drv.setMode(DRV2605_MODE_INTTRIG); // default, internal trigger when sending GO command

drv.selectLibrary(1);
drv.setWaveform(0, 84);  // ramp up medium 1, see datasheet part 11.2
drv.setWaveform(1, 1);  // strong click 100%, see datasheet part 11.2
drv.setWaveform(2, 0);  // end of waveforms
```

To further ease the use of this feature, a generalised and customised method called "playLibraryEffects" is added to the library, below is its implementation:

```
// This function implements a easy waveform sequencer. It takes in
// an arbitrary list of library effects, and play them in sequence.
void Adafruit_DRV2605::playLibraryEffects(std::vector<int> & effects) {
    for (unsigned int i = 0; i < effects.size(); i++) {
        setWaveform(i, effects[i]);
        if (i >= 7) break;
    }
    setWaveform(effects.size(), 0);
    go();
}
```

Alternatively, vibration patterns can be directly designed by applying analogue control. This allows more design freedom and is very straightforward. Playing vibration pattern in this way is called Real-Time Playback (RTP) mode [37]. It directly takes in 8-bit digitised voltage values to control the vibration strength, as mentioned earlier, higher supplier voltage implies higher vibration intensity. A library example [36] is shown below:

```
// The array "rtp" contains several pairs of amplitude and delay,
// in each iteration of the loop below, the output voltage of the driver
// will be first set to the amplitude value, and then will be held
// for the period that specified by the delay value.
// A real-time vibration pattern is thus formed by all those pairs.

uint8_t rtp_index = 0;
uint8_t rtp[] = {
  0x30, 100, 0x32, 100,
  0x34, 100, 0x36, 100,
  0x38, 100, 0x3A, 100,
  0x00, 100,
  0x40, 200, 0x00, 100,
  0x40, 200, 0x00, 100,
  0x40, 200, 0x00, 100
};

if (rtp_index < sizeof(rtp)/sizeof(rtp[0])) {
  drv.setRealtimeValue(rtp[rtp_index]);
  rtp_index++;
  delay(rtp[rtp_index]);
  rtp_index++;
}
```

```
24. else {
25.   drv.setRealtimeValue(0x00);
26.   delay(1000);
27.   rtp_index = 0;
28. }
```

Similar to library effect's case, for convenience, a customised function is added to the original library.

```
1.  // This function implements a easy real time amplitude sequencer. It takes in
2.  // an arbitrary list of amplitudes and a fixed delay, and play them in sequence.
3.  void Adafruit_DRV2605::playRealTimeEffects(std::vector<int> & effects, int time) {
4.      for (unsigned int i = 0; i < effects.size(); i++) {
5.          setRealtimeValue(effects[i]);
6.          delay(time);
7.      }
8.      setRealtimeValue(0x00);
9.  }
```

These two functions are very handy and will be the basis of intuitive ways of pattern design.

2.1.5 DISPLAY

2.1.5.1 LED STRIP (ABANDONED)

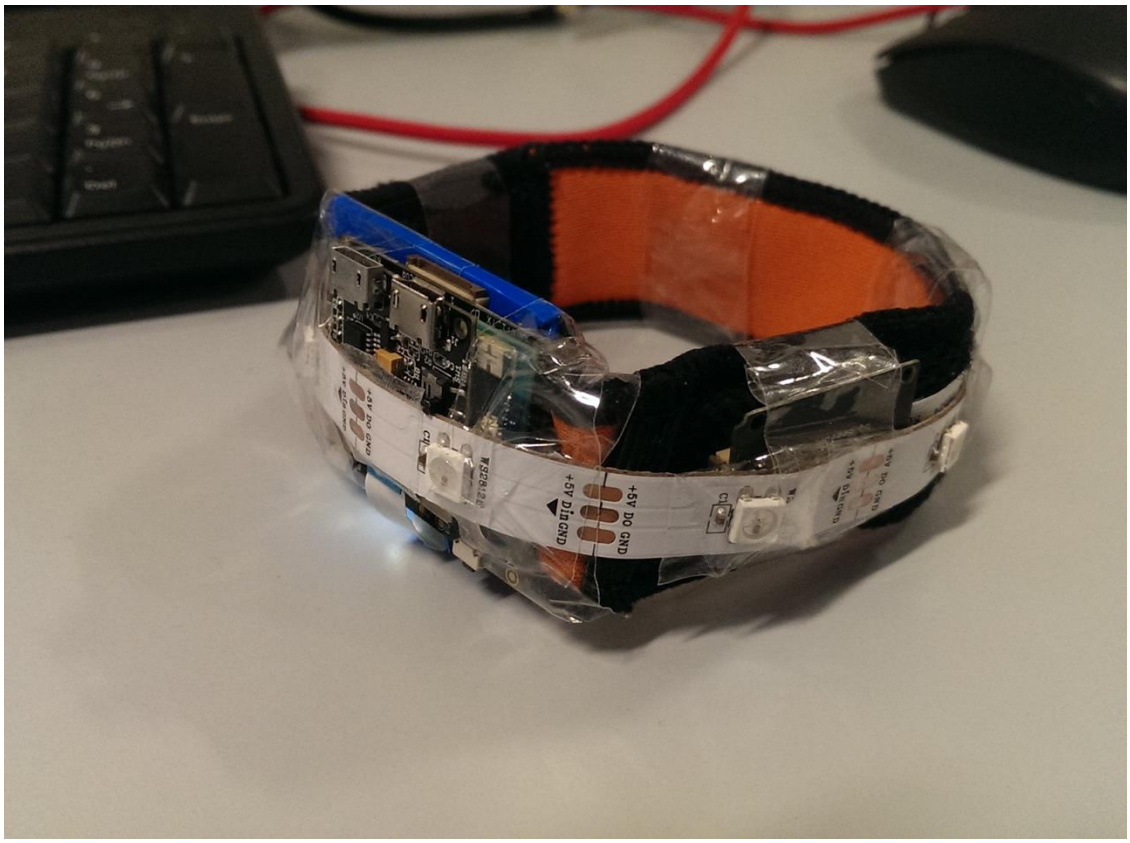


Figure 2-22: LED Strip Wraps Around the Watch Strap

The initial design uses a strip of Light Emitting Diodes (LEDs) to support light pattern alerts. It proves to be a good design choice since it can produce strong lights that could alert those who are partially sighted. Figure 2-22 shows an abandoned design with the LED strip. Each LED has integrated control circuit WS2812b. With the use of its Arduino library [39], the LEDs on the strip is addressable and able to play complex patterns like rainbow colour effects.

Nevertheless, it has been temporarily abandoned for two reasons. One is that it is found difficult to route it in the mechanical design. The other reason is that there is a better alternative, which is Organic Light Emitting Display (OLED).

2.1.5.2 OLED DISPLAY

The OLED display is an appealing choice because it can not only provide a light pattern to the users but can also display texts and provides a UI facilitating the developer in testing and debugging. Besides, the OLED has well-known merits of low power consumption, wide colour gamut and wide viewing angle.

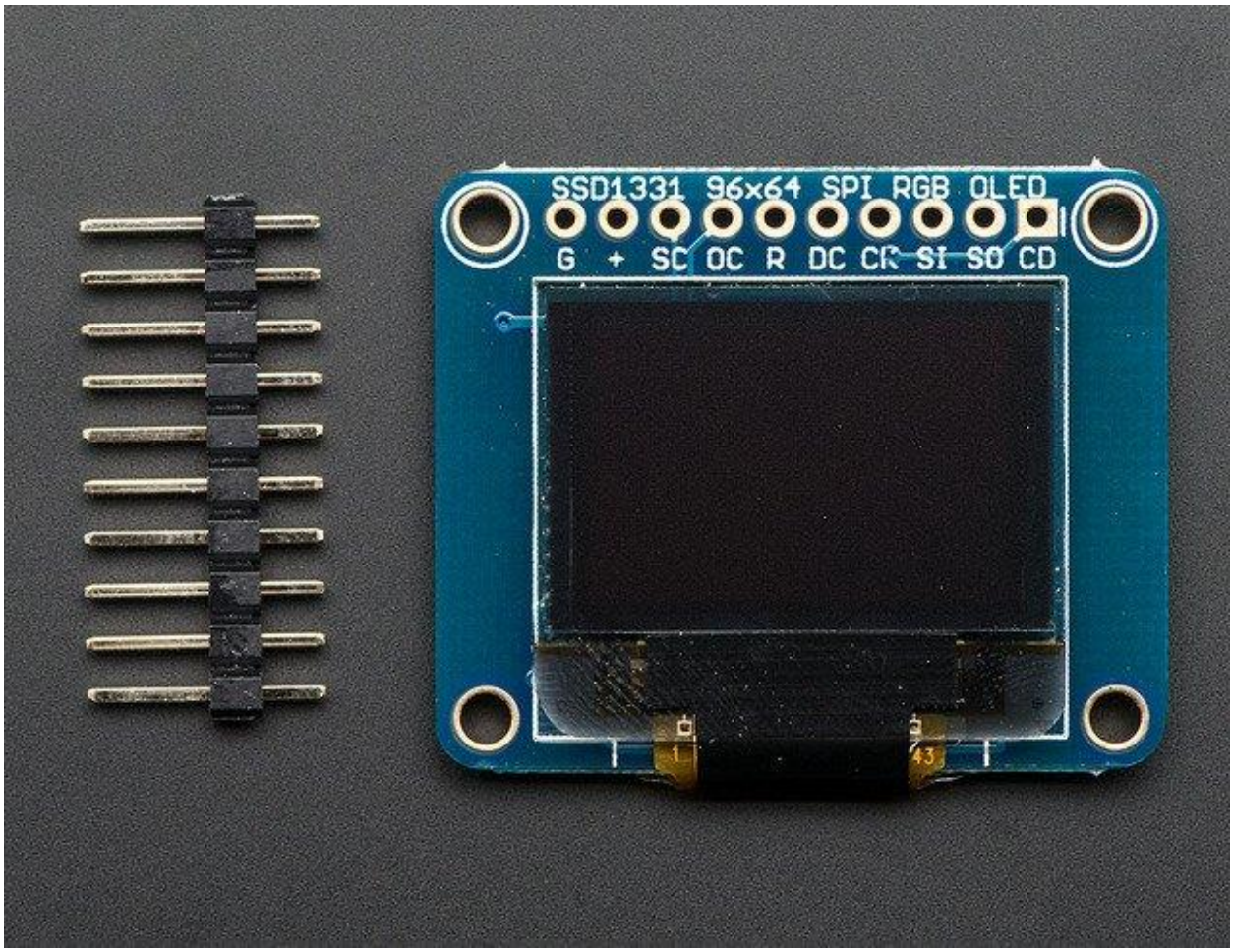


Figure 2-23: Adafruit 0.96" Colour OLED [40]

The specific OLED display was chosen for this project is Adafruit's 0.96" mini colour OLED display. This OLED display consists of a raw colour OLED panel with a resolution of 96 * 64 and an OLED driver SSD1331 [41]. Especially, there is an SD card slot on it, this can be used for loading image to the display, but is not used in this project. The wiring of it with Xadow-Accelerometer (who also breaks out pins for use) is shown in Figure 2-24.

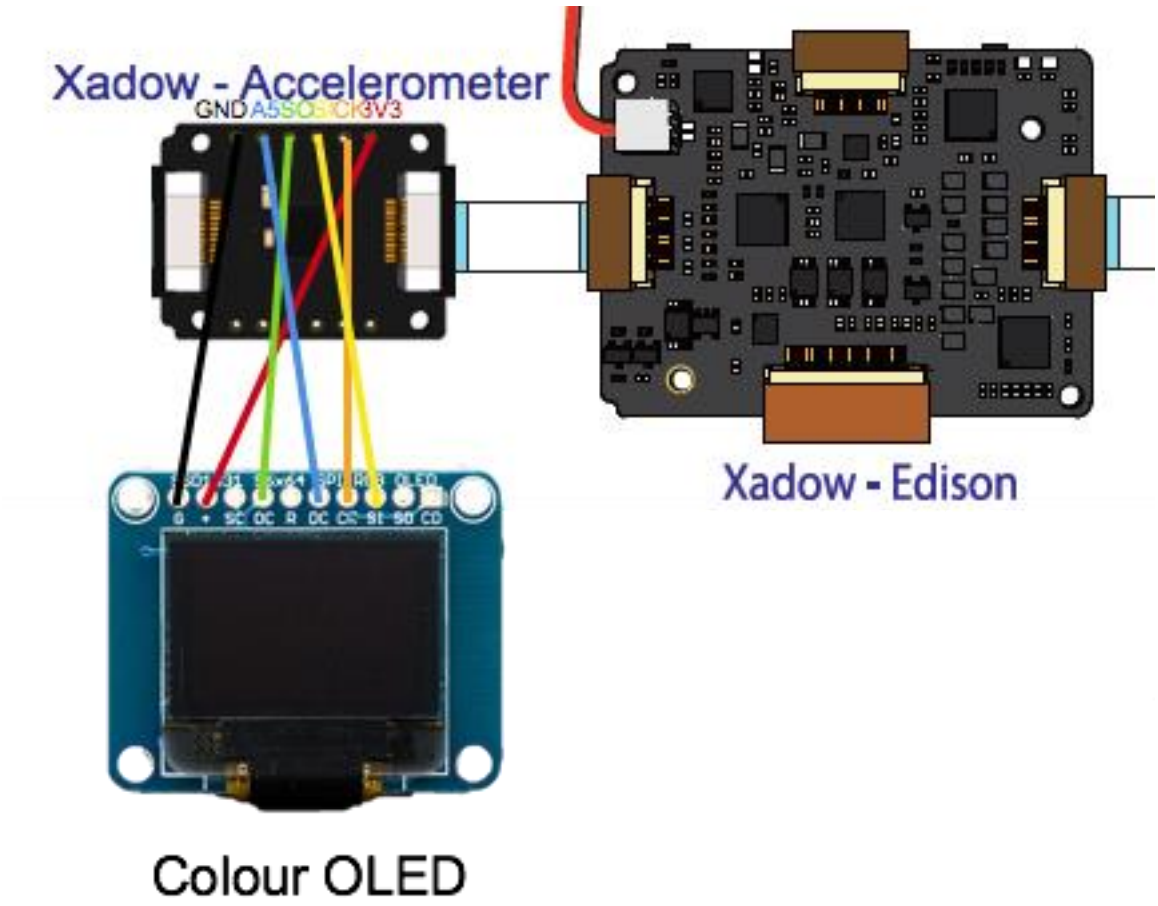


Figure 2-24: Wiring of Colour OLED

As shown above, the pin names on the Xadow-Accelerometer board do not match those of OLED. For clarity, they are further explained in the following table.

Table 2-2: Pin Name Mapping between Colour OLED and Xadow-Accelerometer

Pin name on OLED PCB	Pin name on Xadow - Accelerometer	Description
GND (G)	GND	Ground
VCC (+)	3V3	3V3
SDCS (SC)	skip	SD Card Chip Select

OCS (OC)	MISO	OLED Chip Select
RST (R)	skip	Reset
D/C (DC)	A5	Data/Command
SCK (CK)	SCK	Serial Clock
MOSI (SI)	MOSI	Master Out Slave In
MISO (SO)	skip	Master In Slave Out
CD (CD)	skip	SD Detect

The pin mapping in Arduino Code:

```
1.  // OLED Pin Name    // Pin Name on Xadow-Accelerometer
2.  #define cs      10   // MISO
3.  #define dc      12   // A5
4.  #define mosi    11   // MOSI
5.  #define sclk    13   // SCK
6.
7.  // Actually only two pins needed
8.  // because of the use of dedicated SPI protocol
9.  SSD1331 oled = SSD1331(cs, dc);
```

2.1.5.3 SPI PROTOCOL

Serial Peripheral Interface (SPI) is another popular protocol for low-level device communication. Normally, devices communicate on four SPI lines [34] as Figure 2-25, namely:

- SCLK: a synchronous clock bus line allowing the master to send a clock signal to all slaves.
- SSn: A chip select line to select the slave the master talk to
- MOSI: A data line stands for Master Out Slave In, on which the master send data to the slaves.
- MISO: A data line stands for Master In Slave Out, on which the slaves send data to the master.

The data transfer behaviour can be described by Figure 2-26. It is relatively simple compared to I2C communication. When a slave is selected, its SS line is brought low and the clock line is activated for the use of both master and slave. At the meantime, the master is generating an output at the MOSI line and sampling at the MISO line [34].

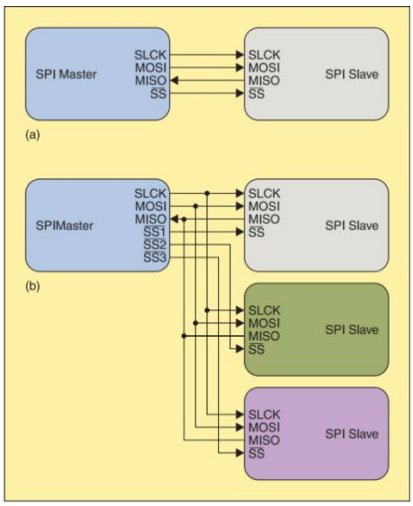


Figure 2-25: SPI Protocol

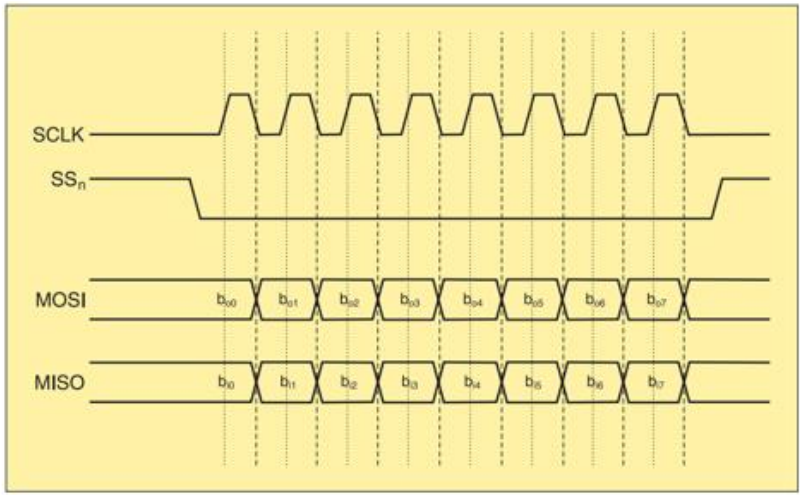


Figure 2-26: Timing Diagram of SPI Protocol

However, in some systems, there are dedicated SPI buses. If used, there is no need to use four GPIOs as a full SPI protocol. The OLED is in such case so that it can be driven solely by Chip Select and Data/Command lines. To sends a byte of command or data the device, it firstly brings the Data and CS line low in the order, then sends the command by using standard SPI library's transfer function. Once it is done, the CS line is brought back to high. This is demonstrated by the following two functions in the device's library (which is customised library based on Adafruit's and Seeed Studio's SSD1331 driver libraries [42] [43]):

```
1.  // To send data, first bring dc and cs low,
2.  // then transfer data, finally bring cs high
3.  void SSD1331::_sendCmd(uint8_t c)
4.  {
5.      digitalWrite(_dc, LOW);
6.      digitalWrite(_cs, LOW);
7.      spiwrite(c);
8.      digitalWrite(_cs, HIGH);
9.  }
10.
11. void SSD1331::spiwrite(uint8_t c) {
12.     // _mosi is 0 when using hardware SPI
13.     // then directly using SPI library funtion
14.     if (!_mosi) {
15.         SPI.transfer(c);
16.         return;
17.     }
18.
19.     // Otherwise, generate SCK and write MOSI following the timing diagram
20.     int8_t i;
21.
22.     digitalWrite(_sck, HIGH);
23.
24.     for (i = 7; i >= 0; i--) {
25.         digitalWrite(_sck, LOW);
26.         if (c) {
27.             digitalWrite(_mosi, HIGH);
28.         }
29.         else {
```

```
            digitalWrite(_mosi, LOW);
        }
        digitalWrite(_sck, HIGH);
    }
}
```

2.1.5.4 Features and APIs

With correct sequences of commands specified in SSD1331's datasheet [41], various settings for the OLED can be done, including setting data format (RGB or BGR), flipping display orientation, changing scanning method, enabling display scrolling, and so on. The following display initialisation is a good example of it:

```
void SSD1331::init(void)
{
    pinMode(_dc, OUTPUT);

    if (_sck) {
        pinMode(_sck, OUTPUT);
        pinMode(_mosi, OUTPUT);
    }
    else {
        // using the hardware SPI
        SPI.begin();
        SPI.setDataMode(SPI_MODE3);
    }

    pinMode(_cs, OUTPUT);
    digitalWrite(_cs, LOW);

    _sendCmd(CMD_DISPLAY_OFF);          //Display Off
    _sendCmd(CMD_SET_CONTRAST_A);       //Set contrast for color A
    _sendCmd(0x91);                     //145
    _sendCmd(CMD_SET_CONTRAST_B);       //Set contrast for color B
    _sendCmd(0x50);                     //80
    _sendCmd(CMD_SET_CONTRAST_C);       //Set contrast for color C
    _sendCmd(0x7D);                     //125
    _sendCmd(CMD_MASTER_CURRENT_CONTROL);//master current control
    _sendCmd(0x06);                     //6
    _sendCmd(CMD_SET_PRECHARGE_SPEED_A);//Set Second Pre-change Speed For ColorA
    _sendCmd(0x64);                     //100
    _sendCmd(CMD_SET_PRECHARGE_SPEED_B);//Set Second Pre-change Speed For ColorB
    _sendCmd(0x78);                     //120
    _sendCmd(CMD_SET_PRECHARGE_SPEED_C);//Set Second Pre-change Speed For ColorC
    _sendCmd(0x64);                     //100
    _sendCmd(CMD_SET_REMAP);            //set remap & data format
    _sendCmd(0x60);                     //0x60: Not flipped and RGB
    _sendCmd(CMD_SET_DISPLAY_START_LINE);//Set display Start Line
    _sendCmd(0x0);
    _sendCmd(CMD_SET_DISPLAY_OFFSET);   //Set display offset
    _sendCmd(0x0);
    _sendCmd(CMD_NORMAL_DISPLAY);       //Set display mode
    _sendCmd(CMD_SET_MULTIPLEX_RATIO);  //Set multiplex ratio
    _sendCmd(0x3F);
    _sendCmd(CMD_SET_MASTER_CONFIGURE); //Set master configuration
    _sendCmd(0x8E);
    _sendCmd(CMD_POWER_SAVE_MODE);      //Set Power Save Mode
    _sendCmd(0x00);                     //0x00
    _sendCmd(CMD_PHASE_PERIOD_ADJUSTMENT);//phase 1 and 2 period adjustment
    _sendCmd(0x31);                     //0x31
    _sendCmd(CMD_DISPLAY_CLOCK_DIV);    //display clock divider/oscillator frequency
    _sendCmd(0xF0);
    _sendCmd(CMD_SET_PRECHARGE_VOLTAGE);//Set Pre-Change Level
    _sendCmd(0x3A);
```

```
52.     _sendCmd(CMD_SET_V_VOLTAGE);         //Set vcomH
53.     _sendCmd(0x3E);
54.     _sendCmd(CMD_DEACTIVE_SCROLLING);    //disable scrolling
55.     _sendCmd(CMD_NORMAL_BRIGHTNESS_DISPLAY_ON);//set display on
56. }
```

Based on basic drawing commands like drawPoint and drawLine, customised functions (shown below) have been added to meet OLED's functional requirements in this project.

```
1.  public:
2.      // Basic
3.      SSD1331(uint8_t cs, uint8_t dc); // using hardware SPI
4.      SSD1331(uint8_t cs, uint8_t dc, uint8_t mosi, uint8_t sck);
5.      void init(void);
6.      void drawPixel(uint16_t x, uint16_t y, uint16_t color);
7.      void drawLine(uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1, uint16_t color);
8.      void drawFrame(uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1, uint16_t outColor, uint16_t fillColor);
9.      void fillRect(uint16_t x, uint16_t y, uint16_t w, uint16_t h, uint16_t fillcolor);
10.     void copyWindow(uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2);
11.     void dimWindow(uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1);
12.     void clearWindow(uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1);
13.     void clearScreen(void);
14.     void setScolling(ScollingDirection direction, uint8_t rowAddr, uint8_t rowNum, uint8_t timeInterval);
15.     void enableScolling(bool enable);
16.     void setDisplayMode(DisplayMode mode);
17.     void setDisplayPower(DisplayPower power);
18.     void _sendCmd(uint8_t cmd);
19.
20.     // High-level
21.     void drawChar(uint8_t ascii, uint16_t x, uint16_t y, uint16_t size, uint16_t color);
22.     void drawString(const char *string, uint16_t x, uint16_t y, uint16_t size, uint16_t color);
23.     void drawBitMap(uint16_t x, uint16_t y, const uint8_t *bit-
    map, uint16_t width, int16_t height, uint16_t color);
24.     void fillScreen(uint16_t color);
25.     void fullScreenFlash(uint16_t color1, uint16_t interval1, uint16_t color2, uint16_t inter-
    val2, uint16_t times);
26.     uint16_t getContrastYIQ(uint16_t color);
27.     void patternWithText(uint16_t color, const char* str);
28.     void patternWithTextAndFlash(uint16_t back-
    ground_color, const char* text, uint16_t on_time, uint16_t off_time, uint16_t times);
29.     void patternWithTextAndFlash(uint16_t back-
    ground_color, const char* text, uint8_t x, uint8_t y, uint8_t size, uint16_t on_time, uint16_t off_time, uint16_
    t times);
30.     void playSequencedColors(std::vector<int> & effects);
31.     void playSequencedColors(std::vector<int> & effects, const char* text);
32.     void playSequencedColors(std::vector<int> & ef-
    fects, const char* text, uint8_t x, uint8_t y, uint8_t size);
```

For example, function "getContrastYIQ" can generate a high contrast text colour according to the background colour by using a thresholding method; function "patternWithTextAndFlash" can display coloured background with specified texts and flashing effects; function "playSequencedColors" can play a sequence of colours just like the vibration waveform sequencer.

2.1.6 ACCELEROMETER

The accelerometer taken from Xadow Wearable Kit contains a 16-bit digital accelerometer IC, ADXL345, from Analog Device [44]. Acceleration in X, Y and Z directions can be read using its Arduino library [45] like following:

```
1.  double xyz[3];
2.  double ax,ay,az;
3.  adxl.getAcceleration(xyz);
4.  ax = xyz[0];
5.  ay = xyz[1];
6.  az = xyz[2];
```

Apart from measuring acceleration, it has off-of-shelf single/double tap detection features, offering them as interrupt resources. This feature is particularly helpful for the visually impaired. It provides an easy much easier way for them to interact with the device, comparing with buttons. Because the visually impaired might have trouble in finding a button on the device, while they will encounter no difficulty in knocking the device to attempt an active input.

Here is an example of how to detect them:

```
1.  byte interrupts = adxl.getInterruptSource();
2.  if (adxl.triggered(interrupts, ADXL345_DOUBLE_TAP)) {
3.     // Double tap detected
4.  }
5.  else if (adxl.triggered(interrupts, ADXL345_SINGLE_TAP)) {
6.     // Single tap detected
7.  }
```

However, tap detection is unwanted when the device is vibrating because hence the vibration may trigger some tasks and result in mistakes. To avoid this, the interrupts can be temporarily switched off as following:

```
1.  adxl.setInterrupt(ADXL345_INT_SINGLE_TAP_BIT, 0);
2.  adxl.setInterrupt(ADXL345_INT_DOUBLE_TAP_BIT, 0);
```

It is also able to set the sensitivity, direction and other parameters of tapping detection during setup:

```
1.  //look of tap movement on this axes - 1 == on; 0 == off
2.  adxl.setTapDetectionOnX(0);
3.  adxl.setTapDetectionOnY(0);
4.  adxl.setTapDetectionOnZ(1);
5.
6.  //set values for what is a tap, and what is a double tap (0-255)
7.  adxl.setTapThreshold(50); //62.5mg per increment
8.  adxl.setTapDuration(15); //625us per increment
9.  adxl.setDoubleTapLatency(80); //1.25ms per increment
10. adxl.setDoubleTapWindow(200); //1.25ms per increment
```

These allows more precise control for tapping action detection and lowers the possibility of false judgement.

2.1.7 BLUETOOTH

The Bluetooth communication is a fast wireless communication protocol. It is desired in this project, particularly for tasks like sending decision signals from a Base-Computer or other device like a mobile phone to the device or sending pattern sequences to the device to preview and test their actual effects. The figure below is such example.

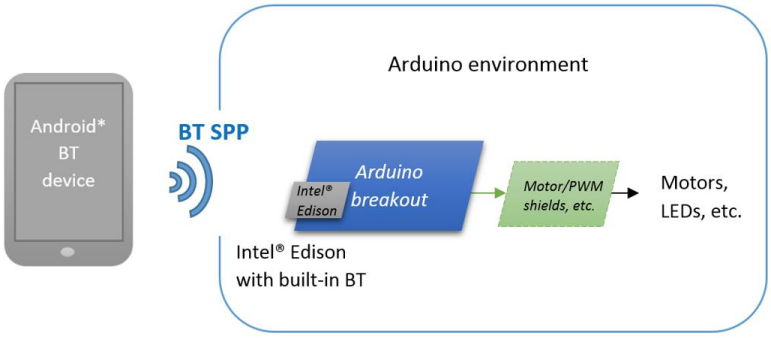


Figure 2-27: Intel Edison and Android Device Communicate through BT SPP [46]

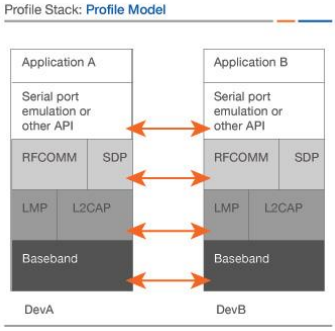


Figure 2-28: Bluetooth SPP Model [47]

To utilise the built-in Bluetooth 4.0 module for Arduino programming, a Bluetooth service running in the background of Intel Edison's Linux operating system needs first to be activated. This is written in Python and can be obtained from [46]. What it does is basically to create a Bluetooth Serial Port Profile (SPP) to receive serial messages sent from paired Bluetooth device, say an Android phone with an SPP app [46]. As its name indicates, SPP, whose model is shown in Figure 2-28, is a serial port emulated by Bluetooth [47]. This allows text messages to be transferred. The received texts are then written to a First In First Out (FIFO) file which is also created by that Python script. The Arduino program then can constantly read out a message from this FIFO to see whether there are new instructions. An example of sending a message from a BT SPP App to Intel Edison's serial port monitor is shown below.

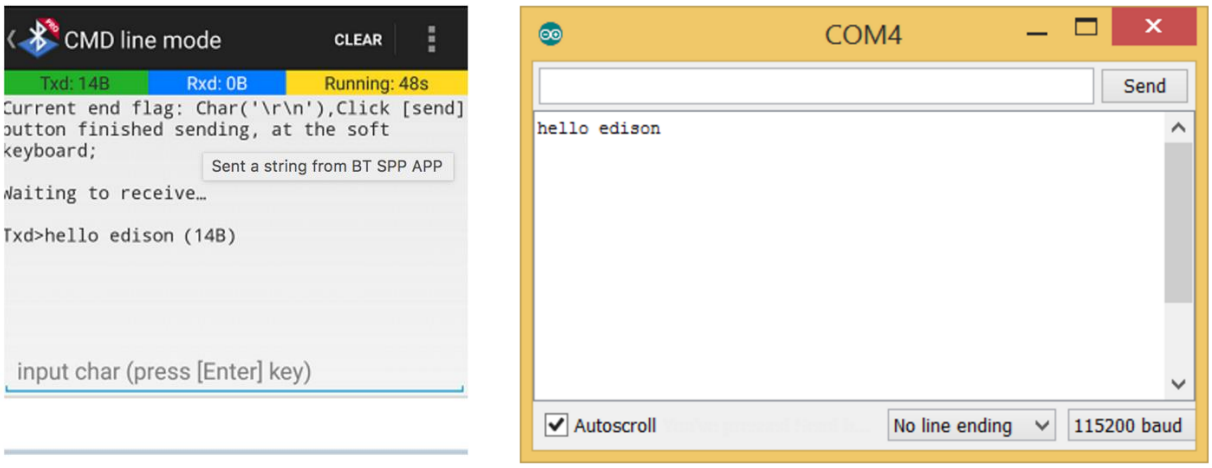


Figure 2-29: Message Sent from BT SPP App Appears in Arduino Serial Monitor [46]

A copy of Intel Edison Bluetooth SPP Arduino Library can also be obtained from [46]. However, the library needs several modifications because the original library opens and reads the file in blocking mode. Rest of the program has to wait until these actions are completed. If there is no file or no message to be read, the program will halt forever, which is unacceptable. To tackle this, they are changed to unblocking mode:

```
1.  // Original open method
2.  //_fd1 = ::open(_pipeName1, O_RDONLY);
3.  // Modified open method:
4.  _fd1 = ::open(_pipeName1, O_RDONLY | O_NONBLOCK);
5.
6.  // Original read method
7.  //ssize_t size = ::read(_fd1, _buf, MAX_BUF - 1);
8.  // Modified read method
9.  int flags = fcntl(_fd1, F_GETFL, 0);
10. fcntl(_fd1, F_SETFL, flags | O_NONBLOCK);
11. ssize_t size = ::read(_fd1, _buf, MAX_BUF - 1);
```

A further problem of the original Bluetooth solution is that it does not support a bidirectional communication. A fundamental idea for building a bidirectional Bluetooth communication is to modify the original Python script. Similar to the reading-in process, a FIFO dedicated to writing out could be created. This FIFO would be opened as a read-only file in the Python script and be opened as a write-only file in the SPP Arduino library. The reading and writing process would run simultaneously as two different Linux system task threads. A new Python script provided by [48] and a newly added SPP Arduino library function "write" have been tested and are able to write a message back to the paired device with no fault.

```
1.  // Write to the output FIFO which can be read by
2.  // the paired device
3.  void Intel_Edison_BT_SPP::write(const char * output) {
4.      if (_fd2 == -1)
5.          open();
```

```
6.
7.      int flags = fcntl(_fd2, F_GETFL, 0);
8.      fcntl(_fd2, F_SETFL, flags | O_NONBLOCK);
9.      Serial.print("To android: ");
10.     Serial.println(output);
11.     ::write(_fd2, output, 1); // just 1 byte
12. }
```

However, another problem occurs after doing so. The Bluetooth service becomes unstable and is not always successfully opened when rebooting the system. It is also found at the later stage that, currently, an echo from the device via Bluetooth is not particularly important, since the device has the other way (Wi-Fi) to interact with the device within the system. As a result, this feature is temporarily disabled for the sake of stability.

2.1.8 Wi-Fi

Although lacking connection speed and robustness when compared with Bluetooth, Wi-Fi as a wireless communication technology has a unique advantage in its coverage and having it is a basic requirement for accessing the Internet and realising the crowdsourcing idea.

Standard Arduino Wi-Fi library [49] can be used for Intel Edison to connect to the Internet. Below is an example of connecting to Wi-Fi within a limited time, since endlessly trying to connect to Wi-Fi is also not acceptable. It also illustrates how OLED can be used for testing.

```
1.  void connectWifi() {
2.      sprintf(buffer, "Connecting Wifi");
3.      Serial.println();
4.      oled.clearScreen();
5.      oled.drawString(buffer, 0, 0, 2, COLOR_WHITE);
6.      // Check for the presence of the shield:
7.      if (WiFi.status() == WL_NO_SHIELD) {
8.          Serial.println("WiFi shield not present");
9.          // don't continue:
10.         while (true);
11.     }
12.
13.     // Check firmaware version
14.     String fv = WiFi.firmwareVersion();
15.     if (fv != "1.1.0")
16.         Serial.println("Please upgrade the firmware");
17.
18.     timestamp = millis();
19.     // Attempt to connect to Wifi network:
20.     while ((status != WL_CONNECTED) && ((millis() - timestamp) < 30000)) {
21.         sprintf(buffer, "Attempting to conncect to SSID: %s", ssid);
22.         Serial.println(buffer);
23.         oled.drawString(buffer, 0, 32, 1, COLOR_WHITE);
24.         // Connect to WPA/WPA2 network. Change this line if using open or WEP net-
    work:
25.         status = WiFi.begin(ssid, pass);
```

```
    }

    // Confirm connected
    if (status == WL_CONNECTED) {
        sprintf(buffer, "Wifi Connected");
        Serial.println(buffer);
        oled.clearScreen();
        oled.drawString(buffer, 0, 0, 2, COLOR_WHITE);
        delay(1000);
        oled.clearScreen();
    }
    else {
        connection_failure = true;  // set failure flag
        sprintf(buffer, "Wifi Not Connected");
        Serial.println(buffer);
        oled.clearScreen();
        oled.drawString(buffer, 0, 0, 2, COLOR_WHITE);
        delay(1000);
        oled.clearScreen();
        return;
    }
}
```

2.1.9 LOCAL STORAGE

When the user is experiencing networking problem and is not able to access the cloud database, a local storage with offline data will render the whole system much more robust.

Local storage of the Intel Edison module can be exploited by using the standard Arduino SD card library. Only a small modification needs to be done in the original library file SD.cpp (located at directory: **C:\...\[arduinoIDE]\hardware\intel\i686\libraries\SD\src\SD.cpp**), in order to ensure all file processing will be performed only in the internal storage. The code snippet is shown below:

```
// The original sd path
// const char* SD_MOUNT_PATH = "/media/sdcard";

// The path after modification, pointing to the internal storage
const char* SD_MOUNT_PATH = "/home";
```

2.1.10 BATTERY

The Li-ion battery taken from Xadow Wearable Kit for Intel Edison can supply 500 mAh electricity at 3.7 V. It is difficult to calculate the device total power consumption in use because it is subject to the actual pattern being played. However, the duration of the device's battery life can be estimated in following three ways:

1 If assuming the system is in idle state, the system is estimated to last for:

$$Power\ consumption\ of\ Intel\ Edison = 330mW$$

$$Battery\ Life\ in\ Idle\ State = \frac{3.7V \times 500mAh}{330mW} = 6.167\ hours$$

2 If assuming a worst case in which the device is constantly playing vibration and light patterns, and vibration motor and OLED are all working at their rated voltage and current (see their datasheets [29] [40]), then it is estimated to last for:

$$Power\ Consumption\ of\ Intel\ Edison, Vibration\ Motor\ and\ OLED = 330mW + 75mA \times 3V + 25mA \times 3.3V = 637.5mW$$

$$Battery\ Life\ in\ Worst\ Case = \frac{3.7V \times 500mAh}{637.5mW} = 2.902\ hours$$

3 If assuming the device is used in a reasonable case in which the vibration and light patterns are activated every 15 minutes and last for 5 seconds (totally 20 seconds every hour):

$$Power\ Consumption\ Every\ Hour = 330mW \times 1h + 225mW \times 20s + 82.5mW \times 20s = 331.7083mWh$$

$$Battery\ Life\ in\ Reasonably\ Used\ Case = \frac{3.7V \times 500mAh}{331.7083mW} = 5.577\ hours$$

The second case is not likely to happen. Then the remaining results show that the major limiting factor of the battery life is still Intel Edison's power consumption. A battery lifetime of 5-6 hours is acceptable because a regular conversation will not exceed this length. Moreover, the device can be switched off to save more power when it is not in use. Overall, if properly used, the device's battery life is expected to be enough for a whole day's use.

2.2 Software Design

2.2.1 Two Working Modes

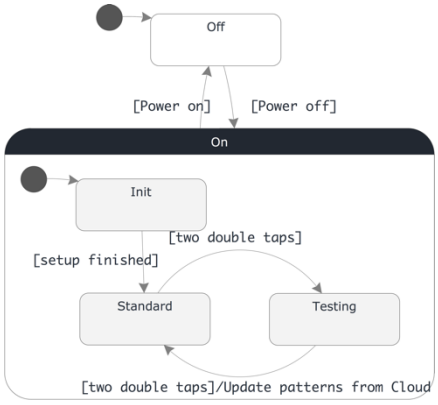


Figure 2-30: Device State Diagram

Because the potential user of the device includes both the visually impaired and pattern designers, the device is designed to be able to work in two different modes. one is "Standard Mode", and the other is "Testing Mode". The two modes are switchable by making consecutive two double taps within four seconds.

2.2.2 Initialisation

When the device is turned on, a quick setup shown below will happen, which will initialise those devices and load local patterns into runtime for use.

```
1.  // Setup at the beginning of the program
2.  void setup() {
3.      // Initialise serial port and oled and display starting page
4.      oled.init();
5.      oled.clearScreen();
6.      oled.drawString("Sentire", 0, 0, 2, COLOR_WHITE);
7.      sprintf(buffer, "Starting...");
8.      Serial.begin(115200);
9.      Serial.println(buffer);
10.     Serial.println();
11.     oled.drawString(buffer, 10, 32, 1, COLOR_WHITE);
12.
13.     // Initialise ohter modules
14.     drv.begin();
15.     drv.selectLibrary(1);
16.     spp.open();
17.     adxl.setup();
18.
19.     //connectWifi();
20.
21.     // Load local patterns to runtime data holder
22.     refreshAllPatternsFromLocalStore();
23.
24.     // Finish setup
25.     Serial.println("Setup Done!");
26.     shortVibrationPulse();
27.     oled.clearScreen();
28. }
```

2.2.3 STANDARD MODE

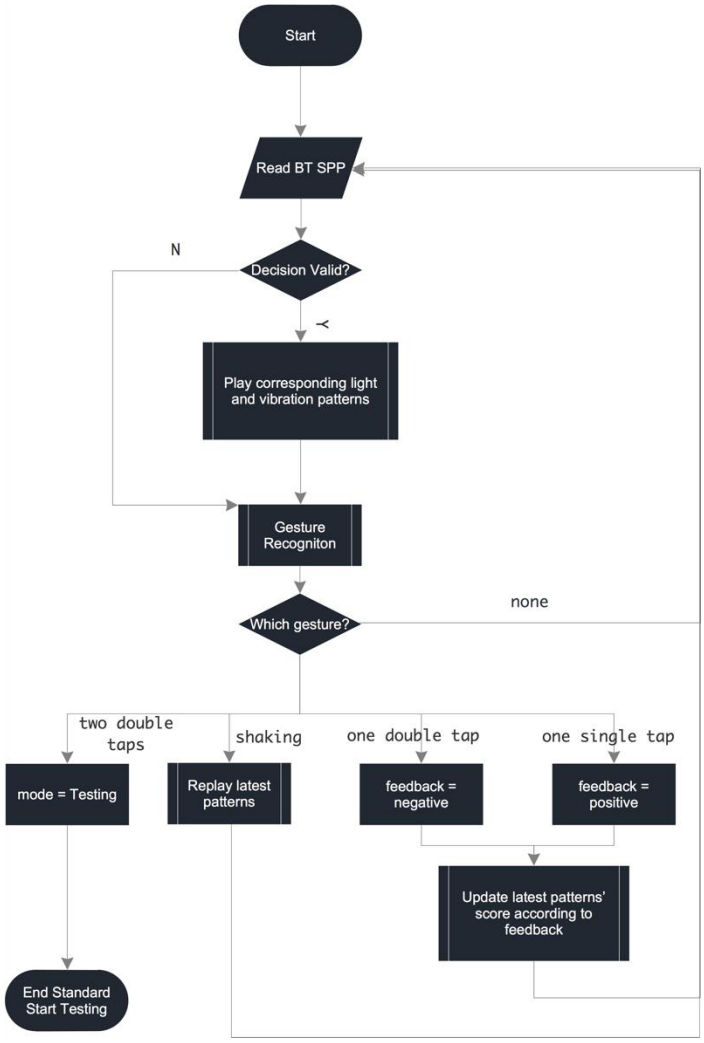


Figure 2-31: Flowchart of Standard Mode

When initialisation finishes, it will automatically enter "Standard Mode". In "Standard Mode", the device is assumed to be used by a visually-impaired user in social contexts. It will play both vibration and light patterns of a certain emotion when a corresponding decision signal is received from a Base-Computer or a mobile phone via Bluetooth as mentioned in Section 1.3. Theses decision signals are encoded in simple numbers to speed processing: 0 – Happy, 1 – Fearful, 2 – Surprised, 3 – Sad, 4 – Disgusted, 5 – Angry. The device will not respond to other messages. The decision number will not be overwritten until new valid one is available from BT SPP. Almost at the same time, the gesture state will be examined, if a clear single tap or a clear double is detected, then update the latest played pattern with a new score (if there is one). The former makes a positive feedback while the later makes a negative one. If double double-tap is detected, then the device mode is changed and it will leave standard mode and enter "Testing Mode". One last gesture could be detected is high-speed shaking, which will lead to the replay of the latest played patterns. This is useful because patterns played for the first time might be partially ignored by the user; then there is a chance left for the user to replay and feel the whole pattern.

2.2.4 Testing Mode

On the other hand, in "Testing Mode", the device is assumed to be under the circumstance that it is being used by a pattern designer or tester. The behaviour of the device in this mode is simpler compared to "Standard Mode". In this case, the device will only listen to a test device such as PC or mobile phone which is now expected to send detailed test patterns. If a test pattern is received, be it vibration type or light type, the device will just play it solely to help the designer generate a preview of pattern they have just designed. This contributes to a fast testing process.

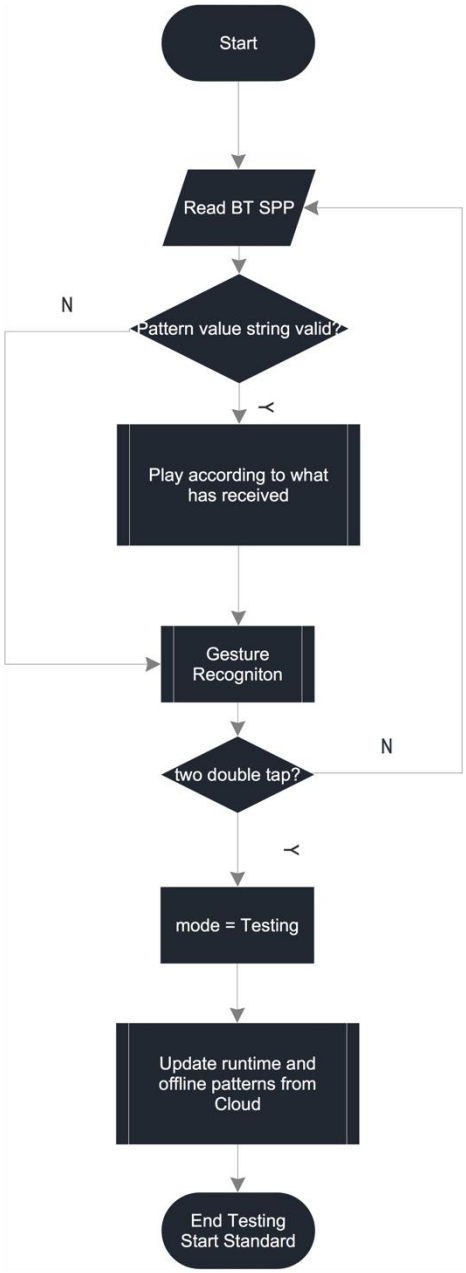


Figure 2-32: Flowchart of Testing Mode

2.2.5 Pattern's Value Format

A pattern's value is specified to have a format as following:

```
1.  // General format
2.  Indicator: Number1, Number2, Number3.......
3.
4.  // Vibration pattern effect based on library effects
5.  1: Effect Number1, Effect Number2, Effect Number3......
6.
7.  // Vibration pattern effect based on real-time amplitudes
8.  2: Amplitude1, Amplitude2, Amplitude3......
9.
10. // Light pattern effect based on colour sequences
11. 3: R1, G1, B1, Delay1, R2, G2, B2, Delay2,......
```

The indicator before the colon is used to distinguish patterns by its type. Specifically, 1 stands for a vibration pattern based on library effects, 2 stands for a vibration pattern based on real-time values, 3 stands for a light pattern.

The numbers after the colon have different meanings in each case. In library-effect based vibration pattern's case, these numbers are the indices of those built-in library effects. If it is a real-time type, then these figures are a series of amplitudes. If it is a light pattern, these numbers follow the format like "R1, G1, B1, Delay1, R2, G2, B2, Delay2…" which defines each colour and its lasting time by using four numbers. Each type needs to be handled by its

particular handler function to realise the actual effect. They have been mentioned before in Hardware Design section, namely “playLibraryEffects()”, “playRealtimeEffects()” and “playSequencedColors()”.

2.2.6 Patten's Data Structure and Format

In order to perform various operations on pattern records like rating, each pattern is not solely defined by its value. Each pattern stored in the cloud has its unique ID, name, emotion, type, value and score. The ID is auto-generated when a record is created; a name is a more human memorisable ID of a pattern; “emotion” defines the type of emotion that pattern is associated with; “type” is defined by its designing methods, who are “Library Vibration (Pattern)”, “Realtime Vibration (Pattern)” and “Light (Pattern)”.

The specific Cloud database used here is Microsoft Azure’s mobile service, and data stored in its table storage is in JavaScript Object Notation (JSON) object), which is useful in passing structured data over the internet in a human readable textual format. The following is an example of a vibration pattern record in the cloud and how does it look like in JSON format:

In Table	id	emotion	name	value	type	score
	0F10BDDE-9E81-43E4-A821..	Angry	a2	1: 39, 38, 37, 36	Library Vibration	1
In JSON	{“id”:“0F10BDDE-9E81-43E4-A821-4BAD1E02580F”,“emotion”:“Angry”,“name”:“a2”,“value”:“1: 39, 38, 37, 36”,“type”:“Library Vibration”, ”,“score”:1}					

Figure 2-33: Pattern Record in Cloud Table and JSON Object Format

Since it is easy to parse and store JSON objects in textual form, patterns held in the runtime and stored in the offline file also keep such format, making them consistent with what stored in the Cloud database.

2.2.7 Access Pattern Data in the Cloud

The device, as mentioned in the previous flowcharts, needs to access the table storage in the Cloud database and perform two basic table operations, namely reading and updating records in the table. Every such operation needs to be done in four steps—sending requests, waiting for responds, reading responds and ending requests. A request in the correct format is required in order to read a correct respond. Its syntax

and format are specified by Microsoft's Azure Mobile Services' REST (Representational State Transfer) APIs [50] . The following is an example illustrating the process of updating a pattern's score (triggered by a negative feedback here). It contains both request to download a pattern and to update its score. To be noticed, the former is needed in order to obtain this pattern's latest score. Otherwise it might end up with the score online being overwritten by a wrong result that is calculated based on an outdated score.

```
Double tap once
:(


connecting...
GET /tables/vibrationpattern?$filter=startswith(emotion,'happy')&$orderby=score+desc&$top=1 HTTP/1.1
Host: songerarduinotest.azure-mobile.net
X-ZUMO-APPLICATION: IBmOdZkslBSsjrCkJeQNvpjHOpTQYr42
Content-Type: application/json
Content-Length: 0


Received:
{"id":"289C45A5-1CE1-4E9B-97F4-29EB9EAF1254","emotion":"Happy","name":"happy2","value":"2: 0, 41, 43, 45, 47, 50, 53, 56, 60, 64, 69, 75, 80, 85, 91, 97, 102,
108, 110, 109, 107, 103, 99, 95, 90, 85, 80, 75, 72, 71, 74, 78, 82, 87, 93, 98, 102, 100, 94, 88, 82, 77, 72, 68, 64, 59, 55, 50, 44, 38, 31, 26, 22, 19, 17,
17, 18, 22, 0, ","type":"Realtime Vibration","score":6}


connecting...
PATCH /tables/vibrationpattern/289C45A5-1CE1-4E9B-97F4-29EB9EAF1254 HTTP/1.1
Host: songerarduinotest.azure-mobile.net
X-ZUMO-APPLICATION: IBmOdZkslBSsjrCkJeQNvpjHOpTQYr42
Content-Type: application/json
Content-Length: 12

{"score": 5}
Received:
{"score":5,"id":"289C45A5-1CE1-4E9B-97F4-29EB9EAF1254"}
```

Figure 2-34: Send Queries and Update Requests and Read Responses

The above example shows how a request is constructed. Generally, it comprises a header and a body.

The header first specifies the purpose of this request by using one of four basic methods of Hypertext Transfer Protocol 1.1 (HTTP 1.1) [51], which are GET, PUT, PATCH and DELETE that corresponds to reading, creating, updating and deleting respectively. It is followed by a Unified Resource Identifier (URI) that is similar to a Unified Resource Locator (URL). In a query request, this URI targets at a specific table entity by first applying a filter to find out patterns of a certain emotion in the table, then by ordering them by their scores in descending order, and finally by looking for the top one. This is querying for the top rated or the most "intuitive" pattern of a certain emotion. In an update request, the pattern wants to be updated will be directly addressed by its unique ID. The reminder of a header specifies the host (URL of the service created), its application key, content type and length of the request body. The body might be empty for a query request while in an update request, the required change(s) needs to be written down there in JSON object format.

After sending a request, the Wi-Fi client will wait for Cloud's response. The response is also in JSON object format. It will be read into the buffer, be printed to the Serial Monitor like the screenshot, and/or be saved for later uses. The query's respond, for example, is the full information of a pattern as described before, whereas, the updating's respond is partial information showing that the score field is updated.

2.2.8 Manage Patterns in Cloud, Runtime, Offline

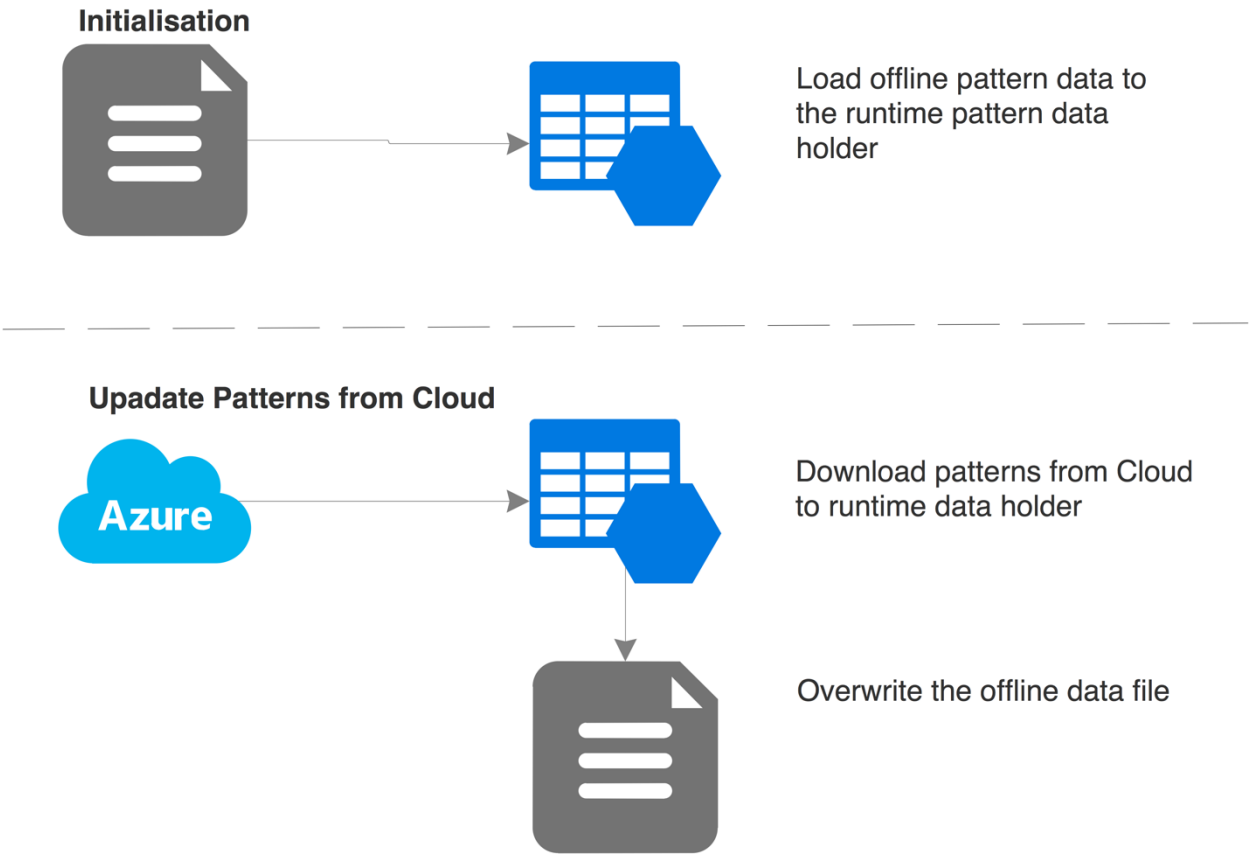


Figure 2-35: Pattern Data Transfer

Pattern data is managed in the ways shown in Figure 2-35. In setup(), it is desired to directly load an offline copy of pattern data to runtime pattern data holder (an array or vector of string type objects) for being quick and robust. When updating patterns from the cloud is needed, it will first overwrite those runtime data, and then the offline file. The second process happens when leaving testing mode for standard mode. In other words, if the user wishes to update patterns from "Standard Mode", two double double-tap are required.

2.2.9 Gesture Recognition

Gesture recognition plays a vital role as discussed so far since it provides an active input to the device to realise both feedbacks and various controls. The uses of each

detected gestures have already been covered in the above context; this section will discuss how these gestures are determined.

The gesture recognition here is expressly the combined tapping and shaking detection. The state machine for detecting them is shown in Figure 2-36.

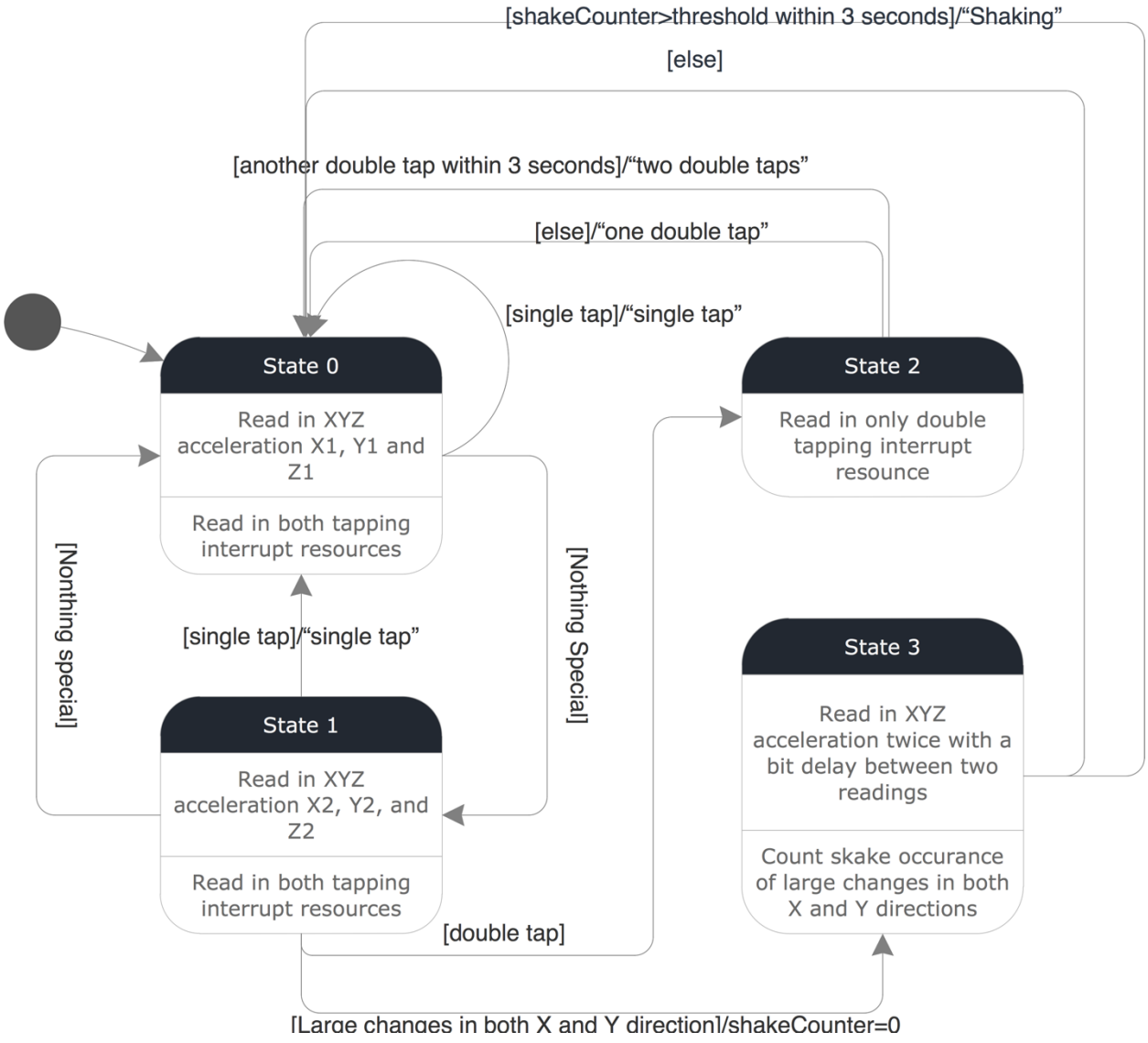


Figure 2-36: State Machine for Gesture Detection

Totally 4 states are involved. State 0 and 1 combines to detect whether there is a significant change in X and Y direction. If so, it will enter State 3 to count the occurrences of such event further. Once the shake counter reaches the threshold within the time, shaking action is confirmed.

Pertaining tapping detection, State 0 and 1 will check both single and double tapping interrupt resources, if a single tap is detected, then it is confirmed. However, if a double tap is found, it needs to go to State 3 for waiting for another possible double

for at most 3 seconds. If there is no second double tap within that time, a single double-tap is confirmed. Otherwise, there is a double double-tap.

2.3 Mechanical Design

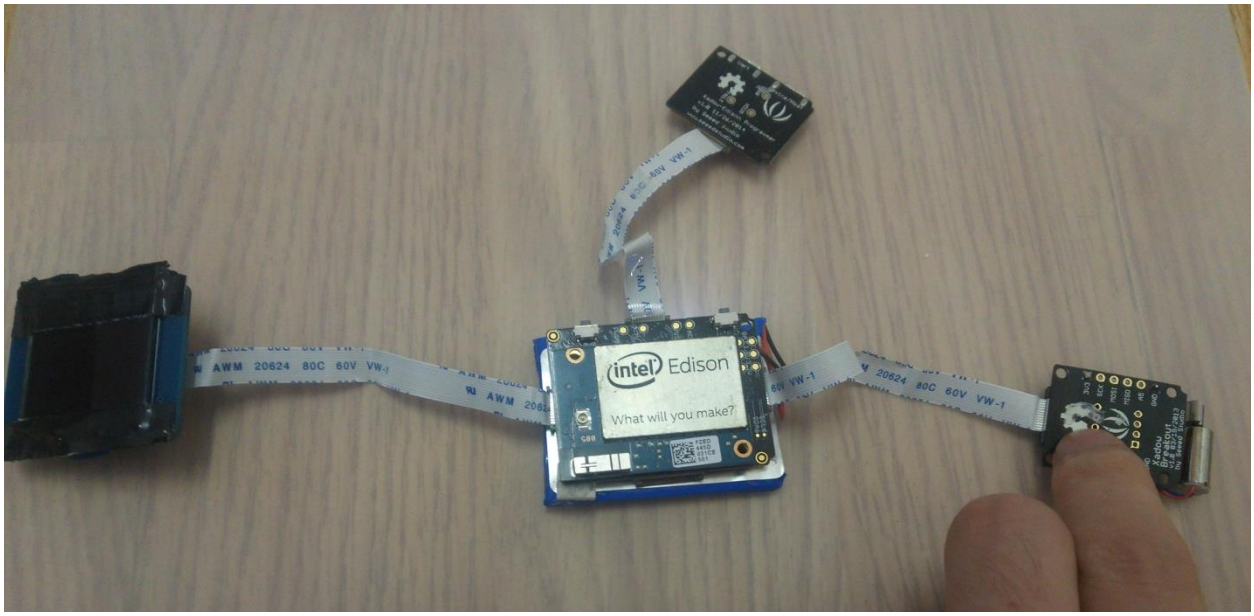


Figure 2-37: Overall Electronics Design

Electronics exposed to external environments are vulnerable to occasional collisions and is almost not possible to test functions like gesture recognition. Therefore, a shell is needed to protect electronics. It has been decided to be laser cut. The design first goes into 3D space, where intuitions about the required dimensions are more easily to be developed. Below are screenshots of the design in TinkerCAD.

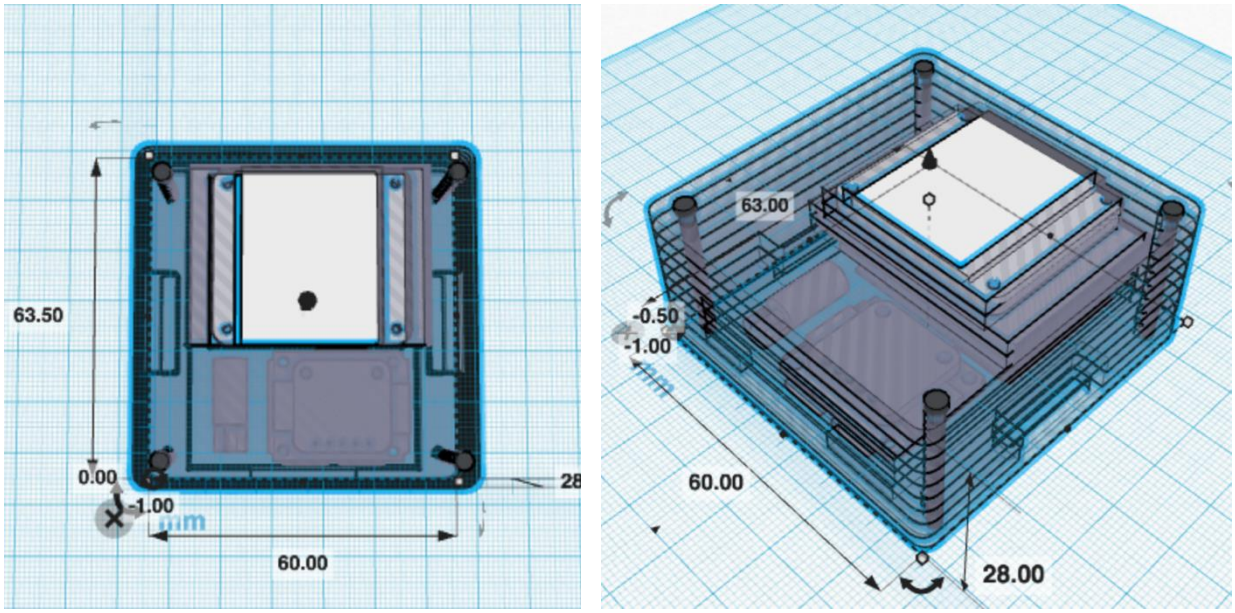


Figure 2-38: Shell Design in TinkerCAD

After that, the design is redrawn in 2D, which is required for the laser cutting machine to operate. The original 2D vector drawing is attached in A.2.

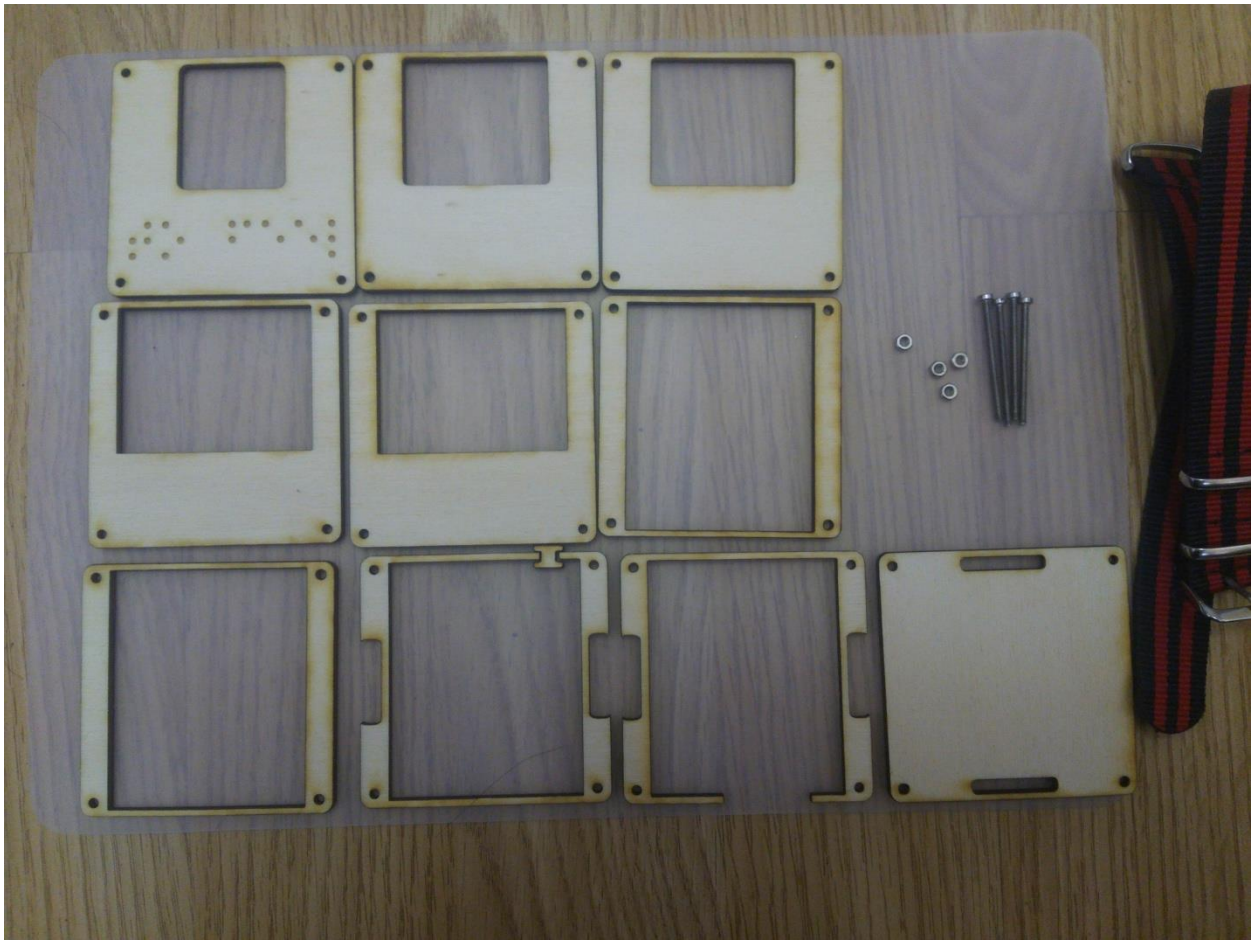
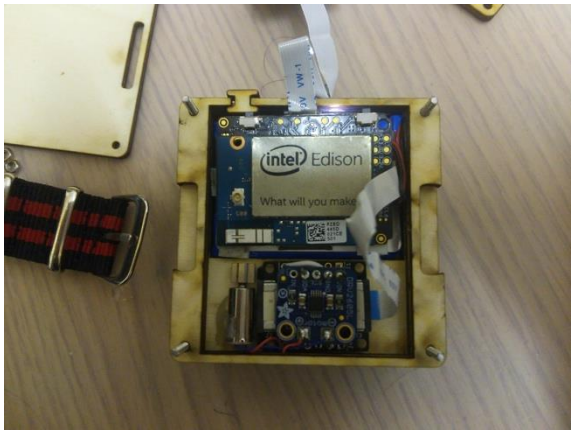
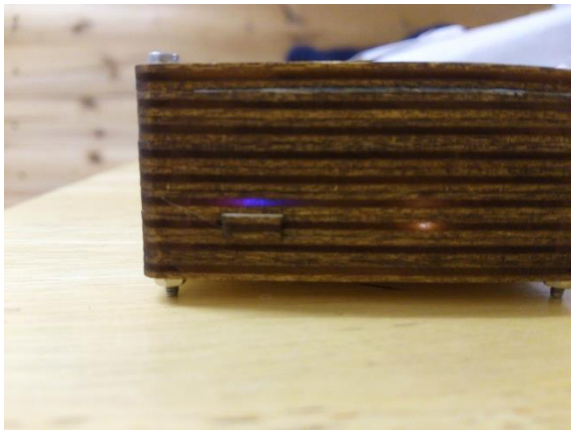


Figure 2-39: Shell pieces and Other materials

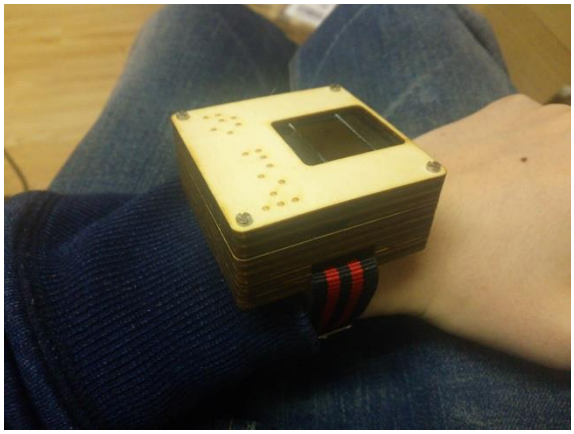
As shown in Figure 2-39, the shell pieces are made of laser grade plywood. The top layer is engraved with British Braille characters meaning "To Feel". The other materials include 4 27mm long M2 metal screws and nuts, and a nylon adjustable watch strap. The assembling process is shown below.



(a) ERM Motor Glued to a Layer



(b) A Button Installed for Power On/Off



(c) LED Indicates Device is On

(d) Device on Wrist

Figure 2-40： Device Assembling

This design is simple, protective, and easy to assemble. However, its size is not minimised and the material lacks flexibility. Other design options include 3D printing, which is also the original plan, but is temporarily abandoned because it takes the long process for it to be completed. A draught design for 3D printing is shown below but is not completed.

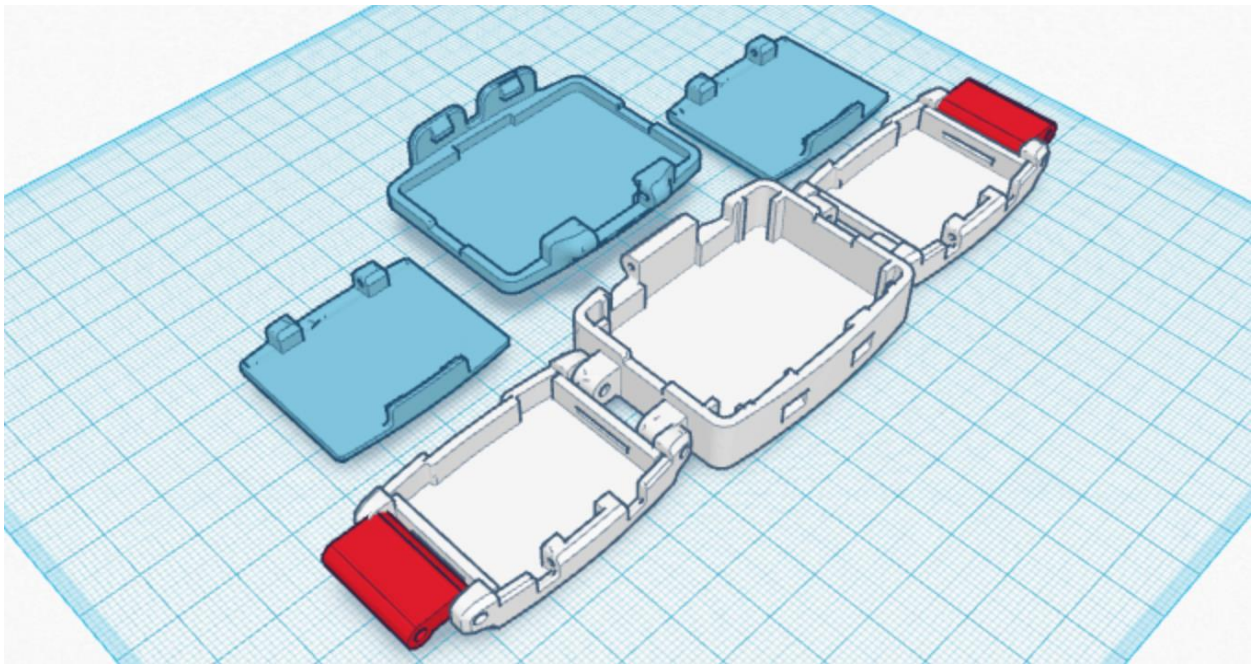


Figure 2-41: Abandoned Shell Design for 3D Printing

Chapter 3 DESIGN OF MOBILE APP

A device alone is not enough for the exploration of feedback pattern intuitiveness, though a lot of useful helper functions has been added to the embedded software design. To make full use of them, an additional device is needed to communicate with the device and to trigger these function dynamically. The mobile phone and a customised app are thus introduced as a device companion. It acts as a vibration/light pattern design and test tool. With it, the pattern designer should be able to design a pattern quickly from scratch by taking advantage of its user-friendly user interfaces (UIs), and perform an instant test for the designed pattern with the device via Bluetooth communication. It will send out pattern values as specified in Section 2.2.5. If a pattern is satisfying, it can be saved and uploaded to the cloud database. Each saved pattern can be rated later inside the app, and its score will be updated to the database as well. Since this project has not been integrated into the larger project yet, there needs to be a platform sending decision signals to the device in "Standard Mode" for the purpose of testing and demonstration. This part is merged to the companion mobile app as well. As a result, there is a "demo" page in the app, where pattern designer or tester can select an emotion decision signal and send to the device.

Regarding the specific mobile platform, Android is an ideal option since it has a dominant position in the current smartphone market. Moreover, it is an open-source platform and has a large developer community. A variety of open-source libraries, APIs, tutorials is available online, which makes it ideal development choice. Unfortunately, introducing Android system and mobile phone from basics is beyond the scope of this report and developing Android app requires a certain level of familiarity with Android and JAVA programming experiences, so this section will mainly focus on discussing its functionality, the detailed operations of the designed app and its UI designs.

3.1 DEVELOPMENT SPECIFICATIONS

Table 3-1: General app Information

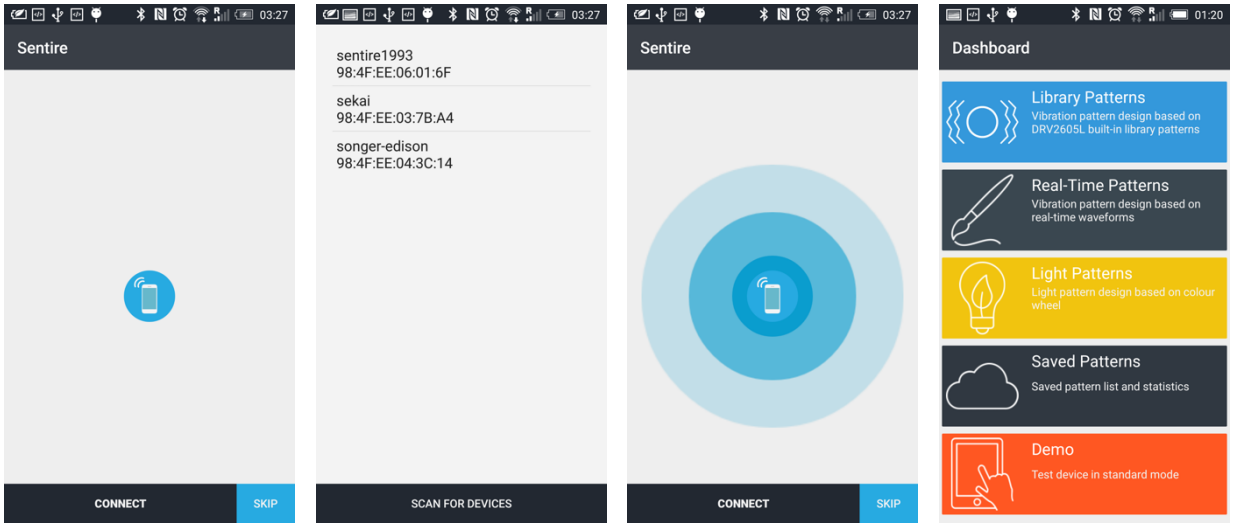
Name of App	**Sentire**
Development Environment	Android Studio 2.0

System Requirements	Android 4.0.4 or above
Hardware Requirements	Android mobile phone or tablet
Connectivity Requirements	• Bluetooth required when testing with the wearable device • Wi-Fi or Mobile Network required when connecting to the Cloud
Third Party Libraries Used	

```
1.  //THIRD-PARTY Open Source Libraries
2.  //Android-BluetoothSPPLibrary: https://github.com/akexorcist/An-
    droid-BluetoothSPPLibrary
3.  compile 'com.akexorcist:bluetoothspp:1.0.0'
4.  //Android Ripple Background: https://github.com/skyfishjy/android-
    ripple-background
5.  compile 'com.skyfishjy.ripplebackground:library:1.0.1'
6.  //EazeGraph: https://github.com/blackfizz/EazeGraph
7.  compile 'com.github.blackfizz:eazegraph:1.2.2@aar'
8.  compile 'com.nineoldandroids:library:2.4.0'
9.  //Color Picker: https://github.com/QuadFlask/colorpicker
10. compile 'com.github.QuadFlask:colorpicker:0.0.10'
11. //Android-Circle Menu: https://github.com/szugyi/Android-CircleMenu
12. compile 'com.github.szugyi:Android-CircleMenu:1.1.1'
13. //Card Library: https://github.com/gabrielemariotti/cardslib
14. compile 'com.github.gabrielemariotti.cards:cardslib-core:2.1.0'
15. compile 'com.github.gabrielemariotti.cards:cardslib-cards:2.1.0'
16. //Microsoft Azure Mobile Services Android Client Li-
    brary: https://azure.microsoft.com/en-gb/documentation/articles/mo-
    bile-services-android-how-to-use-client-library/
17. compile 'com.google.code.gson:gson:2.3'
18. compile 'com.google.guava:guava:18.0'
19. compile 'com.microsoft.azure:azure-mobile-services-android-
    sdk:2.0.3'
20. compile (group: 'com.microsoft.azure', name: 'azure-notifications-
    handler', version: '1.0.1', ext: 'jar')
```

Availability	Available in the Google Play Store: Source code available on Github: https://github.com/songer1993/Sentire

3.2 Connect to Device



(a) (b) (c) (d)

Figure 3-1: UI-Connect to Device

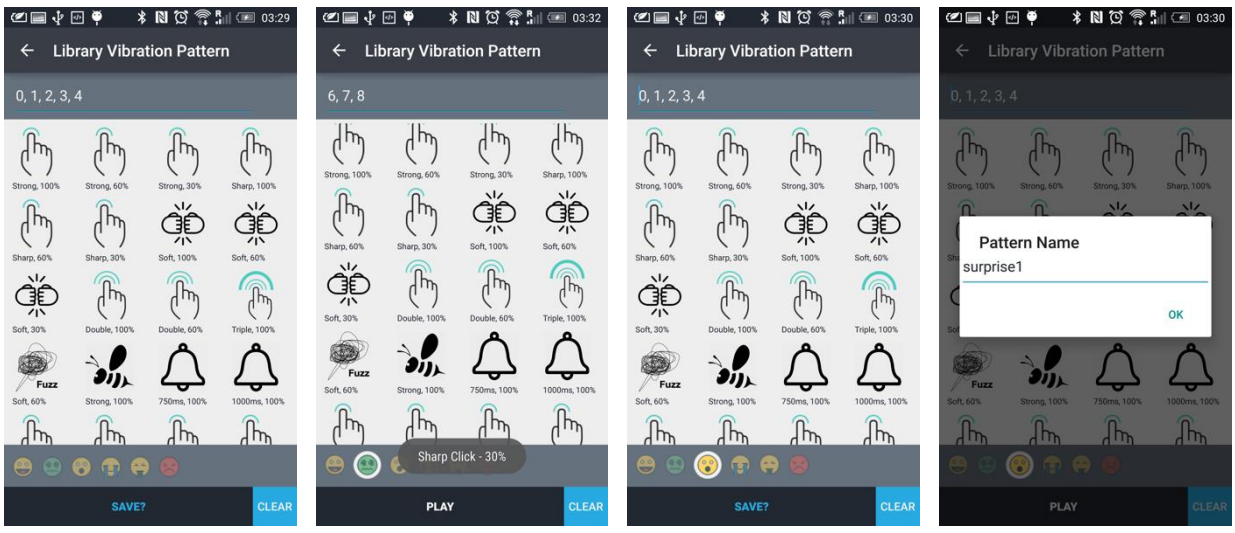
(a) At the start page, the user can click 'Connect' to select device, or skip this step.

(b) The device list where the user can scan and choose the device.

(c) After selecting the device, it will wait for the device to respond, if successful, it will forward into the dashboard.

(d) Dashboard is where the user can choose one task. The user can design patterns based on vibration library, real-time drawing or light. Or the user can see the existing patterns. The user can also test the device in "Standard Mode" with Demo.

3.3 Design Vibration Patterns Based on Library Effects



(a) (b) (c) (d)

Figure 3-2: UI: Design Vibration Patterns Based on Library effects

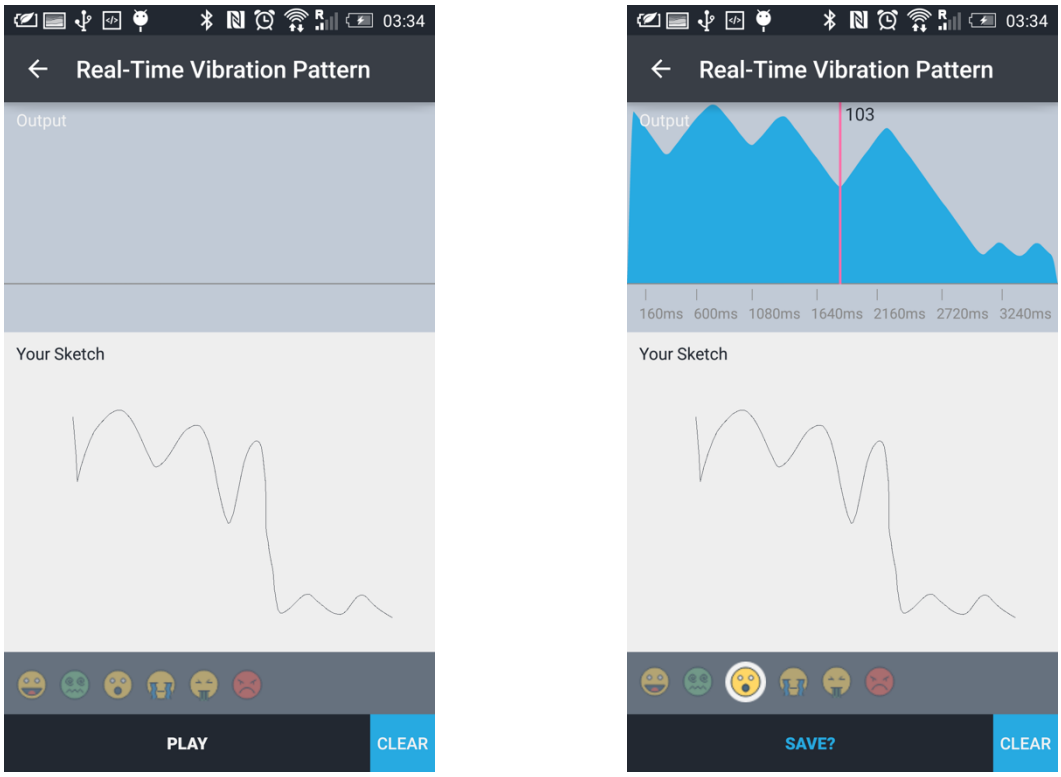
(a) In the library vibration pattern design page, the user can select and insert a library effect from the scrollable panel to the text field by one single tap.

(b) The effect sequence will grow as the user input more effect. A long click on the effect icon will pop up its full name, which is more informative. The "Clear" button can be used to clear the existing sequence.

(c) Once the user is done with inputting, the sequence can be sent to the device by clicking the "Play" button. At the same, time, the play button will change to "Save" button.

(d) If the user is satisfied with this pattern, it can be saved. Its emotion type can be defined by the user by selecting one of face icons. The app will ask the user for a nickname for the designed pattern when the "Save" button is clicked. The user can input a name that is meaningful and memorisable then confirm.

3.4 Design Vibration Patterns Based on Real-Time Effects



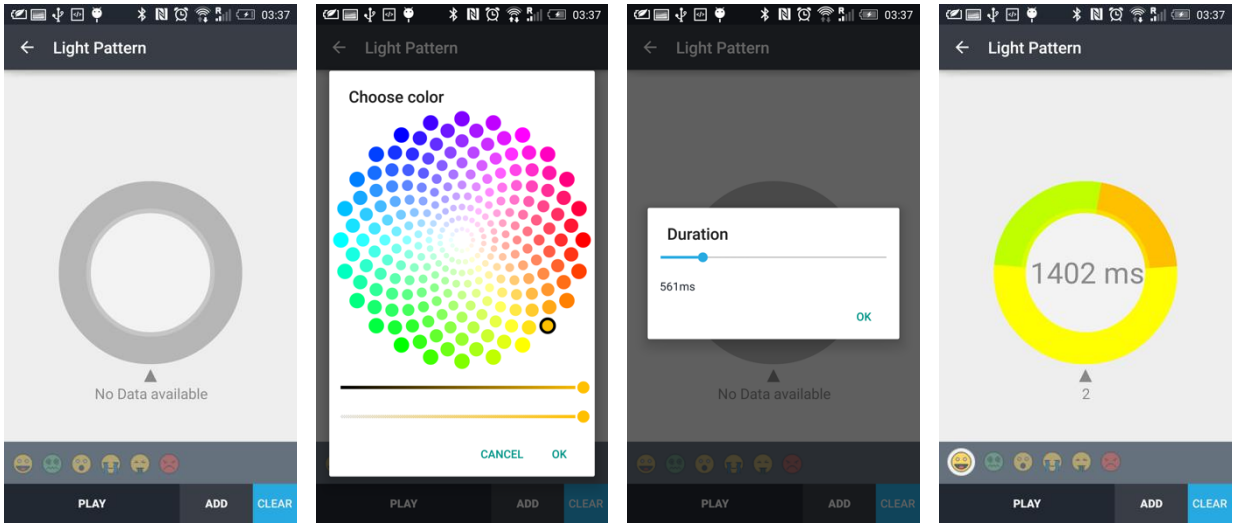
(a) (b)

Figure 3-3: UI-Design Vibration Pattern Based on Amplitudes

(a) Using this design method, the user can draw an arbitrary waveform that is directly related to the vibration intensity with respect to time. The same "Save" and "Clear" buttons apply here.

(b) Similarly, after sketching, the user can play this pattern with the device. Additionally, the upper screen will show a smoothed version of the sketch. This is because it is not reasonable to send all samples from touch display to the device. It will result in a relatively long pattern. Hence, the app is doing a down-sampling process to sample those raw samples. Hence, the waveform becomes smoother.

3.5 Design Light Patterns Based on Colour Sequencer



(a) (b) (c) (d)

Figure 3-4: UI-Design Light Pattern

(a) In light pattern design UI, colour can be added one by one by clicking the "Add" button.

(b) A colour can be selected from this HSV colour picker, which also allows lightness and situation control.

(c) The user then needs to specify the duration of this colour.

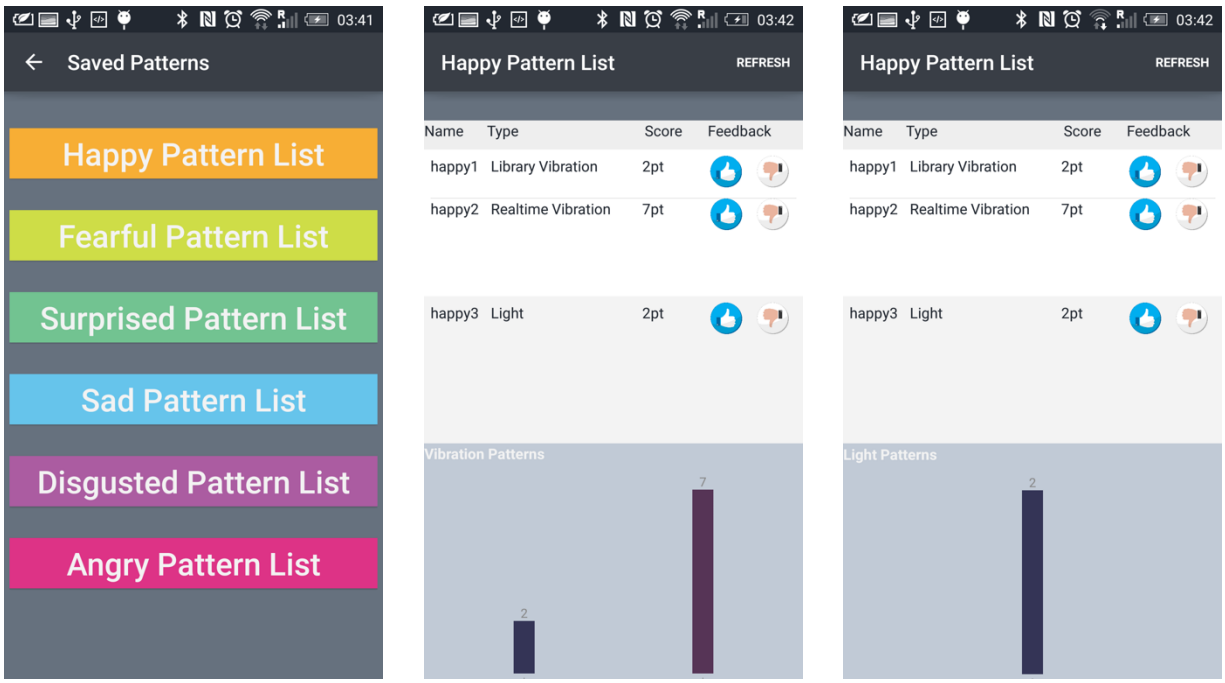
(d) The resulting colour sequence is visualised as pie chart indicating their duration.

3.6 View and Rate Saved Patterns

(a) The user can browse the saved patterns by emotion category.

(b) Taking Happy Pattern List for example, when entering this page, the data from Cloud will be loaded to the upper part screen and they are divided into vibration pattern and light pattern group. Each group view is scrollable. The lower part of UI visualises the statistics about the pattern's scoring. By scrolling, the bar charts of both vibrations, a light pattern can be viewed. Besides, in the UI, the user is allowed to replay the listed patterns by long-clicking. There's also handy thumb

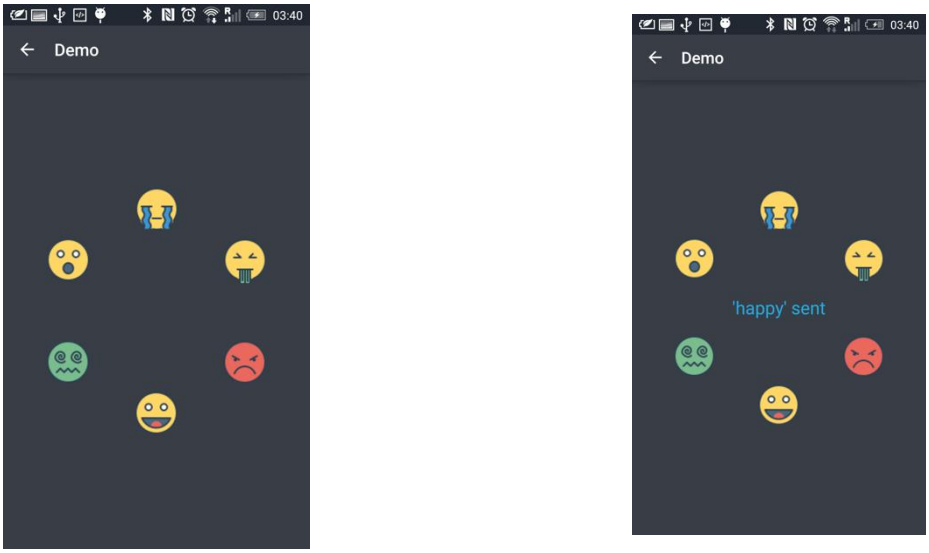
up and thumb down button available for updating one pattern's score. The refresh button is used to update the current list.



(a) (b) (c)

Figure 3-5: UI-View Saved Patterns

3.7 Simulate a Base-Station Computer



(a) (b)

Figure 3-6: UI-Demo

A demo page simply contains six buttons, which will send a number representing one emotion type to the device. This is discussed in the Standard Mode of the device. The use of this page is to emulate a Base-Station computer, that is assumed to send such

decision signals based on its image analysis result. Since this project has yet been connected to the upper part of the parent project, the mobile phone app is also used as a replacement of that part, allowing the device in Standard Mode to be tested.

Chapter 4 System Integration and Testing

To link the designed patterns with the local pattern stored in the device, one last piece of the whole system needs to be filled. This is the cloud database.

The cloud database serves as online storage, bridging the patterns that have been designed and the local patterns stored in the device. It also enables the synchronisation between cloud pattern data and local pattern data of each individual device. In this way, every device can sync with the cloud to obtain the most intuitive patterns (patterns with the highest score in each emotion category).

Creating an own database means a server needs to be built, which is a bit beyond the scope of this project. But fortunately, there are handy third party cloud databases available, like Microsoft Azure, IBM Bluemix. Here Microsoft Azure is used since its system is relatively stable.

It is easy to create a table service for both mobile phone and the device to access by following Microsoft's guides and documentations [52]. The data entity-relationship can be described by Figure 4-1. This is not complicated since now all the potential users will use the same two tables (one stores vibration patterns, and the other stores light patterns). The case will be more complicated in the future when the users might have their own set of tables so that patterns are much more customised for individual users.

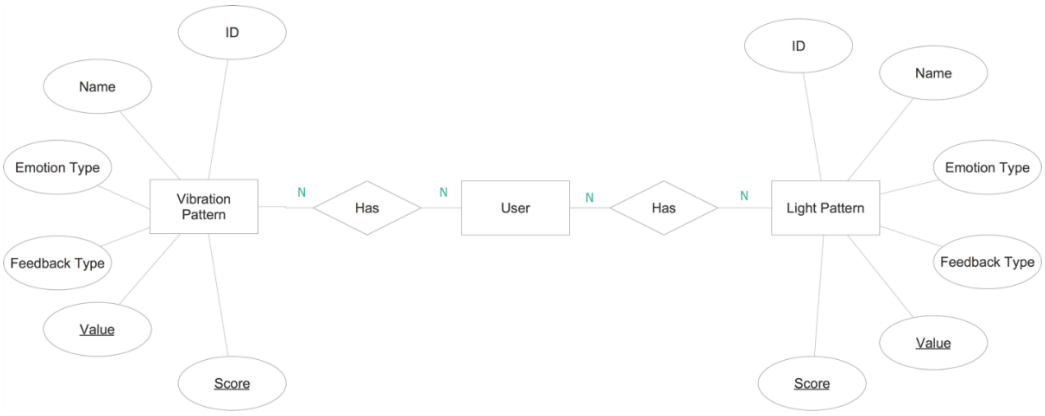


Figure 4-1: Entity-Relationship Diagram

4.1 SYNC TESTING

Figure 4-2: Device in Updating

After updating the device pattern, its internal file is successfully updated by comparing the light patterns in the cloud and the patterns in the local file (Figure 4-3 and Figure 4-4).

lightpattern

BROWSE SCRIPT COLUMNS PERMISSIONS

id	emotion	name	value	type	score
0706E404-D236-40E4-ADD...	Fearful	fear3	3: 128, 255, 0, 989, 23, 255, ...	Light	3
5715A5B9-675B-4B23-8668...	Angry	angry3	3: 255, 23, 110, 1359, 255, 1...	Light	14
62F7C080-B873-4597-ACD...	Happy	happy3	3: 255, 192, 0, 561, 255, 255...	Light	2
8C501D84-DEBA-4621-8FE...	Disgusted	disgust3	3: 45, 66, 99, 1079, 20, 8, 99...	Light	1
CFCE459D-2816-42C9-A6A...	Sad	sad3	3: 116, 238, 255, 1503,	Light	1
F3CC2BD7-065B-4E5D-92D...	Surprised	surprise3	3: 46, 255, 99, 1068, 134, 25...	Light	3

Figure 4-3: Viewing Table Online

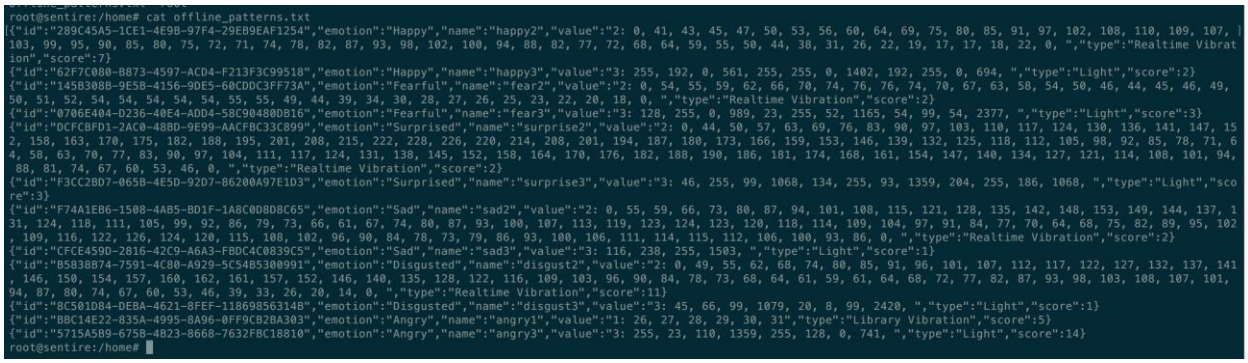


Figure 4-4: Contents in the Offline File

This is proved to be a successful synchronisation. However, it might fail, which depends on the network connection. Hence, the use of the offline file is very beneficial.

Chapter 5 Pathway to Impact

5.1 Current Position in Product Life Cycle

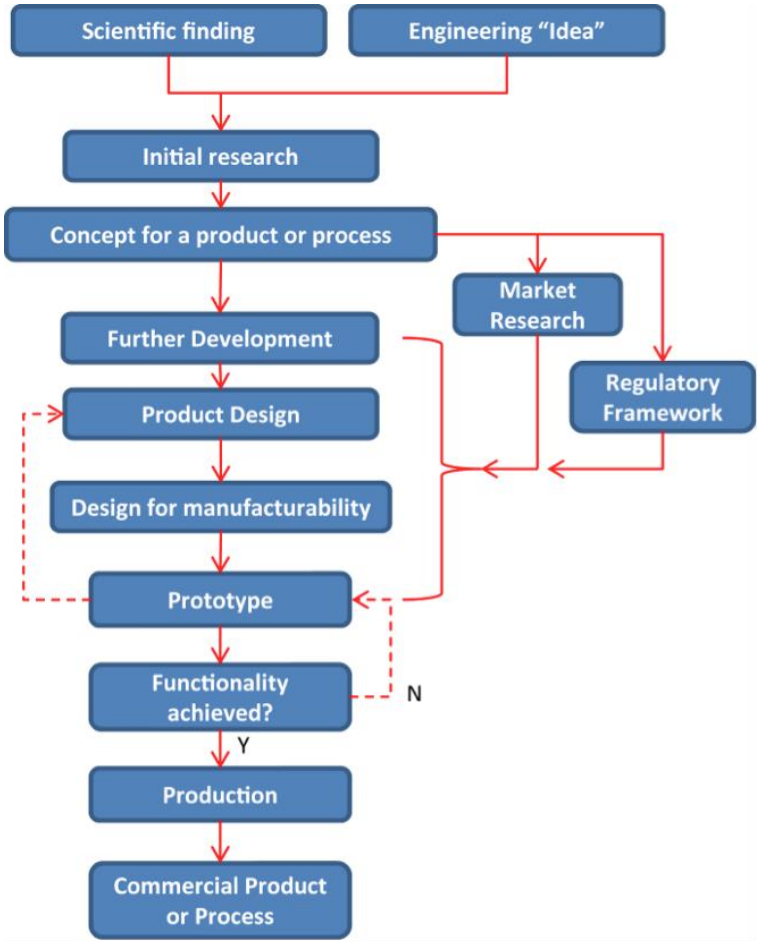


Figure 5-1: Typical Commercial Product Development Flow [53]

This project now is at a position between the initial research and the further development stage in a typical product development flow (Figure 5-1). It started from the initial research of J. Ye et.al [18], and ended up with a prototype device together with its crowdsourcing platform and companion mobile app.

The ultimate aim of this project is to build a novel and mature SAT or SAD that can convert visual cues in the social contexts to intuitive alerting patterns. This is extremely beneficial to a wide range of disadvantaged groups whose majority is the visually impaired. People with deficits in emotion recognition can also take advantage of this technology. Specifically, this technology should contribute to improving the quality of those stakeholder's communication with others. This will help them improve daily and social lives.

To achieve this, further development needs to be done to improve the system's robustness. For example, the database structure could be upgraded to suits a potentially large group of users; the gesture recognition's accuracy could be improved to allow more accurate controls and lower false detection rate. Also, gestures other than those mentioned in the above contexts could be realised. This could be free-fall detection, which is useful to send alert signals to others when a visually impaired

user falls, and so that people can help. Or more complex gesture recognition could be added, such as drawing circle, cross and so on. These complicated gestures could be used to help visually impaired individuals control home appliances, for example.

There are many ways to improve this product or prototype, however, its success will also be subject to the current market's need. This requires cautious market research and user studies.

5.2 Academic Impact

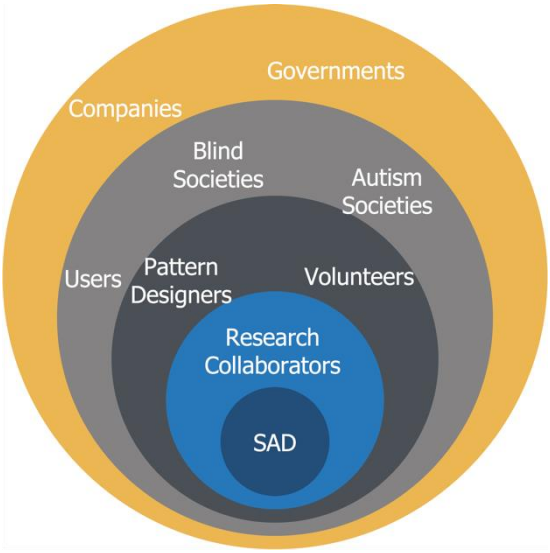


Figure 5-2: Stakeholder Onion Diagram

This project is believed to have strong impacts from academic, social and economic perspectives as the stakeholder onion diagram illustrates. But most of all and in the short run, the predictable impact will be mainly academic.

As described in the introduction, this project is in conjunction with different departments from different Universities. The research result will be interdisciplinary and interscholastic (Figure 5-3), which contributes towards the health of disciplines. Very likely, a related paper will be published in the near future, which will bring other academics' attentions and interests in the related study.

Regarding its content, the most impact will probably be in the psychological area. Because with this device, a user study can be soon carried out. It will be beneficial in investing how external man-made signals can be integrated into human's natural sensory system. The intuitiveness of a feedback pattern thus will also be learnt from it. The result of this research will, in turn, benefits the development of future SATs.

The device itself, is also, a relatively new type of SAD, with a particular focus on conveying emotional information. With the advent of it, this specific topic will be investigated further and relating knowledge is deepening.

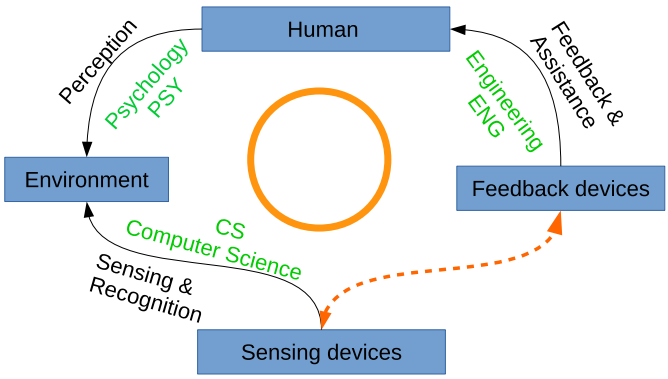


Figure 5-3: A systematic approach to explore machinery and human sensory signals. [18]

5.3 Economic and Social Impact

By this project's nature, it will have significant social impacts since its aim is to improve visually impaired person's live quality. As they become more confident in their social lives by benefiting from this project, the way how the sighted interact with them and their attitude towards them will also have imperceptible positive changes.

In the long run, this project is likely to be commercialised, because of the existence of a vast number of potential users as mentioned earlier and the potential of SAD market.

Chapter 6 CONCLUSION

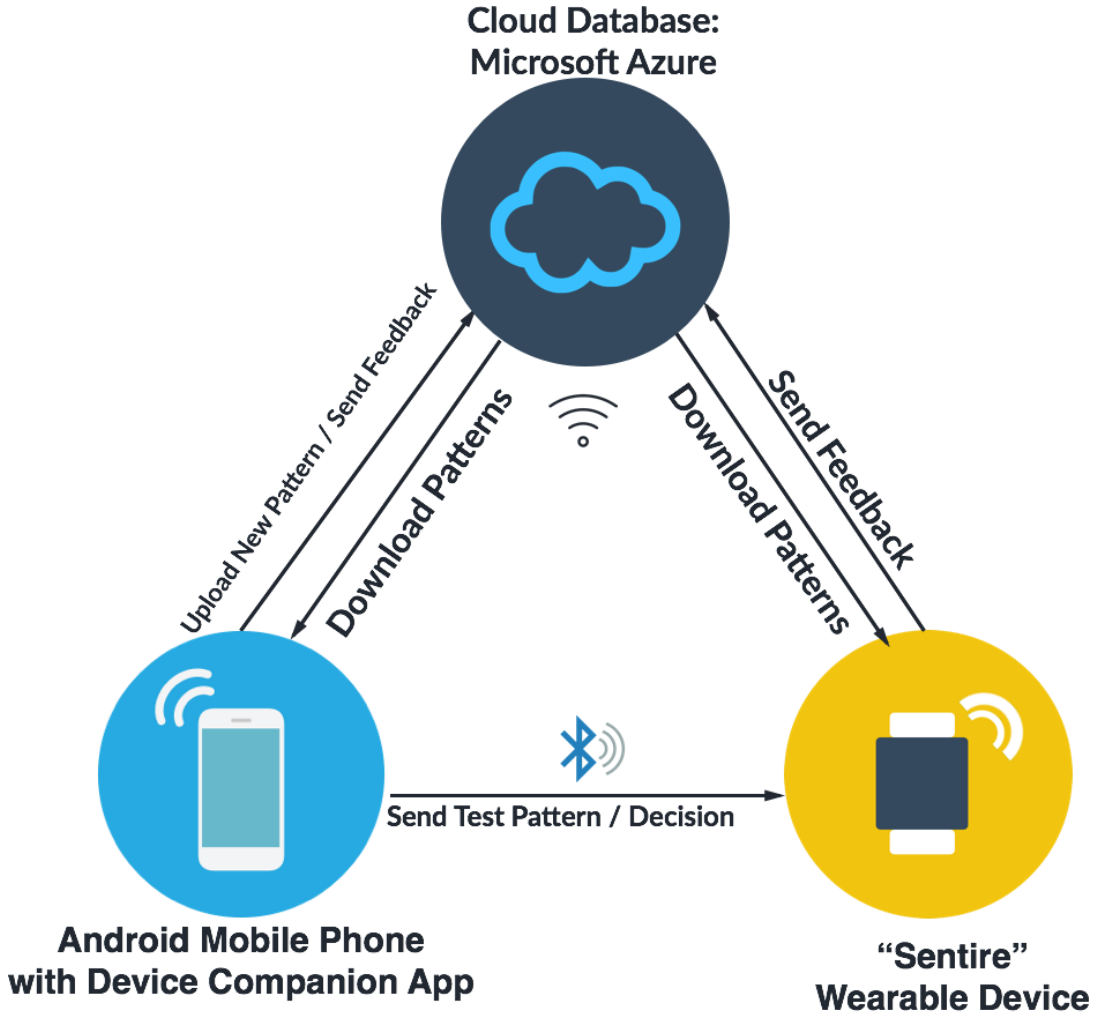


Figure 6-1: "Sentire" (which means "To Feel" in Latin) System Overview

In conclusion, this project has been very successful. A cloud vibration and light pattern design platform, whose core is a sensory assistive device, has been constructed (Figure 6-1). The carefully designed wearable watch has excellent abilities to generate strong alerting vibration and light pattern while is still able to convey subtle information. Moreover, extensive work has been put into the design of a companion mobile app that will significantly ease the design of vibration and light patterns, and thus the advantages of the watch strap can be maximised. By utilising the power of Internet and crowdsourcing idea, the system is able to provide high flexibility and customisability in the alerting pattern design. This has opened avenues for the research which are going to be carried out in the near future.

Chapter 7 ACKNOWLEDGEMENT

I would like to acknowledge my project supervisor, Dr Philip Hands, for his continued kind guidance and support throughout the project. I would like to thank my project examiner, Dr Adam Stokes, for his valuable feedback and advice on the cloud-sourcing app and help in mechanical design and laser cutting of the watch case. Finally, I am grateful to our collaborators, Miss Lauren Murray, Dr Juan Ye, and Dr Ross Goutcher, for their encouragements and appreciations of my creations.

Chapter 8 BIBLIOGRAPHY

[1] A. Mehrabian and S. R. Ferris, "Inference of attitudes from nonverbal communication in two channels," *Journal of consulting psychology,* vol. 31, no. 3, p. 248, 1967.

[2] A. Mehrabian and M. Wiener, "Decoding of inconsistent communications," *Journal of personality and social psychology,* vol. 6, no. 1, p. 109, 1967.

[3] C. R. Darwin, The expression of the emotions in man and animals, 1st Edition ed., London: John Murray, 1872.

[4] M. A. Kassian and B. Hassler, Conversation Peace: Improve Your Relationships One Word at a Time, Nashville, Tennessee : B&H Publishing Group, 2004.

[5] World Health Organisation, "Visual impairment and blindness," WHO Media centre , 1 August 2014. [Online]. Available: http://www.who.int/mediacentre/factsheets/fs282/en/. [Accessed 20 April 2016].

[6] M. B. Harms, A. Martin and G. L. Wallace, "Facial emotion recognition in autism spectrum disorders: a review of behavioural and neuroimaging studies," *Neuropsychology Review,* vol. 20, no. 3, pp. 290-322, 2010.

[7] C. G. Kohler and et al., “Emotion recognition deficit in schizophrenia: association with symptomatology and cognition,” *Biological Psychiatry,* vol. 48, no. 2, pp. 127-136, 2000.

[8] Centres for Diseases Control and Prevention, “Autism Spectrum Disorder (ASD),” Centres for Diseases Control and Prevention, 31 March 2016. [Online]. Available: http://www.cdc.gov/ncbddd/autism/data.html. [Accessed 20 April 2016].

[9] J. Ward and P. Meijer, “Visual experiences in the blind induced by an auditory sensory substitution device,” *Consciousness and Cognition,* vol. 19, no. 1, pp. 492-500, 2010.

[10] G. Jensson, “Haptics as a Substitute for Vision,” in *Assistive technology for visually impaired and blind people*, M. A. Hersh and M. A. Johnson, Eds., Glasgow, Springer Science & Business Media, 2010, pp. 135-160.

[11] P. Bach-Y-Rita and et al., “Vision substitution by tactile image projection,” *Nature,* vol. 221, no. 5184, pp. 963-964, 1969.

[12] S. u. Réhman, “Expressing Emotions through Vibration for Perception and Control,” Umeå University, Umeå , 2010.

[13] S. Krishna and et al., “VibroGlove: an assistive technology aid for conveying facial expressions,” in *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, ACM, 2010.

[14] T. Alexander and et al., “Vibrotactile target saliency,” in *Display Technologies & Applications for Defense, Security, and Avionics II*, Orlando , 2008.

[15] M. Brock and P. O. Kristensson, “Supporting blind navigation using depth sensing and sonification,” in *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, ACM, 2013.

[16] R. Plutchik, "The Nature of Emotions Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice," *American Scientist,* vol. 84, no. 1, pp. 344-350, 2001.

[17] D. McCandless, Information is beautiful, London: Collins, 2009.

[18] J. Ye, P. Hands and R. Goutcher, *Understanding Interaction between Human and Machine Sensory Perception,* the University of St Andrews, the Univeristy of Edinburgh and the University of Stirling: Unpublished, 2015.

[19] H.-Y. Chen and et al., "Tactor localization at the wrist," in *Haptics: Perception, Devices and Scenarios*, Berlin, 2008.

[20] Intel Corporation, "Intel Edison Compute Module," Intel Corportation, 1 Jan 2016. [Online]. Available: http://www.intel.com/content/www/us/en/do-it-yourself/edison.html. [Accessed 20 April 2016].

[21] Intel Coporation, *Intel Edison Module Hardware Guide,* Intel Corporation, 2014.

[22] L. Orsini, "How Intel's Edison Stacks Up Against Arduino And Raspberry Pi," ReadWrite, 11 September 2014. [Online]. Available: http://readwrite.com/2014/09/10/intel-edison-raspberry-pi-arduino-comparison/. [Accessed 20 April 2016].

[23] Seeed Studio, *Xadow Wearable Kit For Intel® Edison,* Shenzhen: Seeed Studio, 2015.

[24] Seeed Studio, "Xadow - Edison," Seeed Studio, 20 May 2015. [Online]. Available: http://www.seeedstudio.com/wiki/Xadow_-_Edison. [Accessed 20 April 2016].

[25] Precision Microdrives, "Eccentric Rotating Mass (ERM) / Pager Motors," Precision Microdrives, 1 January 2016. [Online]. [Accessed 20 April 2016].

[26] Precision Microdrives, "Driving Linear Resonance Vibration Actuators," Precision Microdrives, 1 January 2016. [Online]. Available: http://www.precisionmicrodrives.com/application-notes-technical-

guides/application-bulletins/ab-003-how-to-drive-linear-resonance-actuators-lra-vibrating-motors. [Accessed 20 April 2016].

[27] Precision Microdrives, "Understanding ERM Vibration Motor Characteristics," Precision Microdrives, 1 January 2016. [Online]. Available: http://www.precisionmicrodrives.com/application-notes-technical-guides/application-bulletins/ab-004-understanding-erm-characteristics-for-vibration-applications. [Accessed 20 April 2016].

[28] J. Rockland, "How do devices provide haptic feedback?," Somatic Labs, 7 March 2016. [Online]. Available: https://blog.somaticlabs.io/how-devices-provide-haptic-feedback/. [Accessed 20 April 2016].

[29] Precision Microdrives, "Product Data Sheet Precision Haptic™ 6mm Vibration Motor - 12mm Type," 1 Janury 2016. [Online]. Available: https://catalog.precisionmicrodrives.com/order-parts/product/306-109-6mm-vibration-motor-12mm-type. [Accessed 20 April 2016].

[30] Precision Microdrives, "Integrated Driver Circuits for Vibration Motors," 1 January 2016. [Online]. Available: http://www.precisionmicrodrives.com/application-notes-technical-guides/application-bulletins/ab-017-integrated-driver-circuits-for-vibration-motors. [Accessed 20 April 2016].

[31] Precision Microdrives, "Driving Vibration Motors with Pulse Width Modulation," 1 January 2016. [Online]. Available: http://www.precisionmicrodrives.com/application-notes-technical-guides/application-bulletins/ab-012-driving-vibration-motors-with-pwm. [Accessed 20 April 2016].

[32] Precision Microdrives, "Discrete H-bridge For Enhanced Vibration Control," 1 January 2016. [Online]. Available: http://www.precisionmicrodrives.com/application-notes-technical-guides/application-bulletins/ab-002-discrete-h-bridge-circuit-for-enhanced-vibration-motor-control-haptic-feedback. [Accessed 20 April 2016].

[33] Adadruit Industries, “Adafruit DRV2605 Haptic Controller Breakout,” 17 December 2014. [Online]. Available: https://learn.adafruit.com/adafruit-drv2605-haptic-controller-breakout?view=all. [Accessed 20 April 2016].

[34] F. Leens, “An introduction to I 2 C and SPI protocols,” *Instrumentation & Measurement Magazine, IEEE,* vol. 12, no. 1, pp. 8-13, 2009.

[35] Arduino, “Wire Library,” Arduino, 1 January 2016. [Online]. Available: https://www.arduino.cc/en/reference/wire. [Accessed 20 April 2016].

[36] Adafruit Industries, “Adafruit DRV2605 Library,” Adafruit Industries, 11 April 2016. [Online]. Available: https://github.com/adafruit/Adafruit_DRV2605_Library. [Accessed 20 April 2016].

[37] Texas Instruments, *DRV2605L 2 to 5.2 V Haptic Driver for LRA and ERM With Effect Library and Smart-Loop Architecture,* (Missing): Texas Instruments, 2014.

[38] Precision Microdrives, “Which haptic effects should you use? ” Precision Microdrives, 16 December 2015. [Online]. Available: http://www.precisionmicrodrives.com/tech-blog/2015/12/16/which-haptic-effects-should-you-use. [Accessed 20 April 2016].

[39] Seeed Studio, “Seeed WS2812b Library,” Seeed Studio, 19 April 2015. [Online]. Available: https://github.com/Seeed-Studio/Xadow_Edison_Demos/tree/master/Seeed_WS2812b. [Accessed 20 April 2016].

[40] Adafruit Industries, “0.96" mini Color OLED,” Adadruit Industries, 28 July 2013. [Online]. Available: https://learn.adafruit.com/096-mini-color-oled?view=all. [Accessed 20 April 2016].

[41] Soloman Systech, *SSD1331 Datasheet,* : , 2007.

[42] Adafruit Industries, “Adafruit SSD1331 OLED Driver Library for Arduino,” Adafruit Industries, 1 February 2016. [Online]. Available:

https://github.com/adafruit/Adafruit-SSD1331-OLED-Driver-Library-for-Arduino. [Accessed 20 April 2016].

[43] Seeed Studio, "RGB OLED SSD1331 Library," Seeed Studio, 20 May 2015. [Online]. Available: https://github.com/Seeed-Studio/RGB_OLED_SSD1331. [Accessed 20 April 2016].

[44] Analog Device, *3-Axis, ±2 g/±4 g/±8 g/±16 g Digital Accelerometer ADXL345 Datasheet,* /: Analog Device, 2015.

[45] Seeed Studio, "Accelerometer ADXL345 Library," Seeed Studio, 20 April 2014. [Online]. Available: https://github.com/Seeed-Studio/Accelerometer_ADXL345. [Accessed 20 April 2016].

[46] Intel Corporation, "COMMUNICATE TO ARDUINO CODE WITH YOUR ANDROID* PHONE BY BLUETOOTH SERIAL PORT PROFILE (SPP) ON INTEL® EDISON," Intel Corporation, 19 May 2015. [Online]. Available: https://software.intel.com/en-us/blogs/2015/05/19/communicate-to-arduino-code-with-your-android-phone-by-bluetooth-serial-port. [Accessed 20 April 2016].

[47] Bluetooth, "Serial Port Profile (SPP)," Bluetooth, 1 Januray 2016. [Online]. Available: https://developer.bluetooth.org/TechnologyOverview/Pages/SPP.aspx. [Accessed 20 April 2016].

[48] Niko, "Intel Edison - Bluetooth SPP - Read and Write," StackOverflow, 20 November 2015. [Online]. Available: http://stackoverflow.com/questions/33836313/intel-edison-bluetooth-spp-read-and-write. [Accessed 20 April 2016].

[49] Arduino, "WiFi Library," Arduino, 1 January 2016. [Online]. Available: https://www.arduino.cc/en/Reference/WiFi. [Accessed 20 April 2016].

[50] Microsoft, “Azure Mobile Services REST API Reference,” 3 June 2015. [Online]. Available: https://msdn.microsoft.com/en-GB/library/azure/jj710108.aspx. [Accessed 20 April 2016].

[51] R. Fielding and et al., *Hypertext Transfer Protocol -- HTTP/1.1,* : RFC-2608, 1997.

[52] Microsoft Azure, “Get started with Mobile Services for Android (JavaScript backend),” 8 April 2016. [Online]. Available: https://azure.microsoft.com/en-gb/documentation/articles/mobile-services-android-get-started/. [Accessed 20 April 2016].

[53] University of Edinburgh, *BEng Project Organisation,* Edinburgh: University of Edinburgh, 2016.

[54] Precision Microdrives, “Introduction to Haptic Feedback,” 1 January 2016. [Online]. Available: http://www.precisionmicrodrives.com/haptics-haptic-feedback-vibration-alerting/haptic-feedback-in-detail/experience-and-understand-haptics. [Accessed 20 April 2016].

[55] D. Jones, “Ardjson–Part 3: Command line Json: cURL.exe,” 19 Feburary 2015. [Online]. Available: http://embedded101.com/Blogs/David-Jones/entryid/560/cejsonpart-3-command-line-json-curlexe. [Accessed 20 April 2016].

Chapter 9 APPENDIX

A.1 HAPTIC DRIVER DRV2605L BUILT-IN LIBRARY EFFECTS

EFFECT ID NO.	WAVEFORM NAME	EFFECT ID NO>	WAVEFORM NAME	EFFECT ID NO.	WAVEFORM NAME
1	Strong Click - 100%	42	Long Double Sharp Click Medium 2 – 80%	83	Transition Ramp Up Long Smooth 2 – 0 to 100%
2	Strong Click - 60%	43	Long Double Sharp Click Medium 3 – 60%	84	Transition Ramp Up Medium Smooth 1 – 0 to 100%
3	Strong Click - 30%	44	Long Double Sharp Tick 1 – 100%	85	Transition Ramp Up Medium Smooth 2 – 0 to 100%
4	Sharp Click - 100%	45	Long Double Sharp Tick 2 – 80%	86	Transition Ramp Up Short Smooth 1 – 0 to 100%
5	Sharp Click - 60%	46	Long Double Sharp Tick 3 – 60%	87	Transition Ramp Up Short Smooth 2 – 0 to 100%
6	Sharp Click - 30%	47	Buzz 1 – 100%	88	Transition Ramp Up Long Sharp 1 – 0 to 100%
7	Soft Bump - 100%	48	Buzz 2 – 80%	89	Transition Ramp Up Long Sharp 2 – 0 to 100%
8	Soft Bump - 60%	49	Buzz 3 – 60%	90	Transition Ramp Up Medium Sharp 1 – 0 to 100%
9	Soft Bump - 30%	50	Buzz 4 – 40%	91	Transition Ramp Up Medium Sharp 2 – 0 to 100%
10	Double Click - 100%	51	Buzz 5 – 20%	92	Transition Ramp Up Short Sharp 1 – 0 to 100%
11	Double Click - 60%	52	Pulsing Strong 1 – 100%	93	Transition Ramp Up Short Sharp 2 – 0 to 100%
12	Triple Click - 100%	53	Pulsing Strong 2 – 60%	94	Transition Ramp Down Long Smooth 1 – 50 to 0%
13	Soft Fuzz - 60%	54	Pulsing Medium 1 – 100%	95	Transition Ramp Down Long Smooth 2 – 50 to 0%
14	Strong Buzz - 100%	55	Pulsing Medium 2 – 60%	96	Transition Ramp Down Medium Smooth 1 – 50 to 0%
15	750 ms Alert 100%	56	Pulsing Sharp 1 – 100%	97	Transition Ramp Down Medium Smooth 2 – 50 to 0%
16	1000 ms Alert 100%	57	Pulsing Sharp 2 – 60%	98	Transition Ramp Down Short Smooth 1 – 50 to 0%
17	Strong Click 1 - 100%	58	Transition Click 1 – 100%	99	Transition Ramp Down Short Smooth 2 – 50 to 0%
18	Strong Click 2 - 80%	59	Transition Click 2 – 80%	100	Transition Ramp Down Long Sharp 1 – 50 to 0%
19	Strong Click 3 - 60%	60	Transition Click 3 – 60%	101	Transition Ramp Down Long Sharp 2 – 50 to 0%
20	Strong Click 4 - 30%	61	Transition Click 4 – 40%	102	Transition Ramp Down Medium Sharp 1 – 50 to 0%
21	Medium Click 1 - 100%	62	Transition Click 5 – 20%	103	Transition Ramp Down Medium Sharp 2 – 50 to 0%
22	Medium Click 2 - 80%	63	Transition Click 6 – 10%	104	Transition Ramp Down Short Sharp 1 – 50 to 0%
23	Medium Click 3 - 60%	64	Transition Hum 1 – 100%	105	Transition Ramp Down Short Sharp 2 – 50 to 0%
24	Sharp Tick 1 - 100%	65	Transition Hum 2 – 80%	106	Transition Ramp Up Long Smooth 1 – 0 to 50%
25	Sharp Tick 2 - 80%	66	Transition Hum 3 – 60%	107	Transition Ramp Up Long Smooth 2 – 0 to 50%
26	Sharp Tick 3 – 60%	67	Transition Hum 4 – 40%	108	Transition Ramp Up Medium Smooth 1 – 0 to 50%
27	Short Double Click Strong 1 – 100%	68	Transition Hum 5 – 20%	109	Transition Ramp Up Medium Smooth 2 – 0 to 50%
28	Short Double Click Strong 2 – 80%	69	Transition Hum 6 – 10%	110	Transition Ramp Up Short Smooth 1 – 0 to 50%
29	Short Double Click Strong 3 – 60%	70	Transition Ramp Down Long Smooth 1 – 100 to 0%	111	Transition Ramp Up Short Smooth 2 – 0 to 50%
30	Short Double Click Strong 4 – 30%	71	Transition Ramp Down Long Smooth 2 – 100 to 0%	112	Transition Ramp Up Long Sharp 1 – 0 to 50%
31	Short Double Click Medium 1 – 100%	72	Transition Ramp Down Medium Smooth 1 – 100 to 0%	113	Transition Ramp Up Long Sharp 2 – 0 to 50%
32	Short Double Click Medium 2 – 80%	73	Transition Ramp Down Medium Smooth 2 – 100 to 0%	114	Transition Ramp Up Medium Sharp 1 – 0 to 50%
33	Short Double Click Medium 3 – 60%	74	Transition Ramp Down Short Smooth 1 – 100 to 0%	115	Transition Ramp Up Medium Sharp 2 – 0 to 50%
34	Short Double Sharp Tick 1 – 100%	75	Transition Ramp Down Short Smooth 2 – 100 to 0%	116	Transition Ramp Up Short Sharp 1 – 0 to 50%
35	Short Double Sharp Tick 2 – 80%	76	Transition Ramp Down Long Sharp 1 – 100 to 0%	117	Transition Ramp Up Short Sharp 2 – 0 to 50%
36	Short Double Sharp Tick 3 – 60%	77	Transition Ramp Down Long Sharp 2 – 100 to 0%	118	Long buzz for programmatic stopping – 100%
37	Long Double Sharp Click Strong 1 – 100%	78	Transition Ramp Down Medium Sharp 1 – 100 to 0%	119	Smooth Hum 1 (No kick or brake pulse) – 50%
38	Long Double Sharp Click Strong 2 – 80%	79	Transition Ramp Down Medium Sharp 2 – 100 to 0%	120	Smooth Hum 2 (No kick or brake pulse) – 40%
39	Long Double Sharp Click Strong 3 – 60%	80	Transition Ramp Down Short Sharp 1 – 100 to 0%	121	Smooth Hum 3 (No kick or brake pulse) – 30%
40	Long Double Sharp Click Strong 4 – 30%	81	Transition Ramp Down Short Sharp 2 – 100 to 0%	122	Smooth Hum 4 (No kick or brake pulse) – 20%
41	Long Double Sharp Click Medium 1 – 100%	82	Transition Ramp Up Long Smooth 1 – 0 to 100%	123	Smooth Hum 5 (No kick or brake pulse) – 10%

Figure 9-1: List of Built-In Library Vibration Effect [37]

A.2 MECHANICAL 2D DESIGN DRAWING

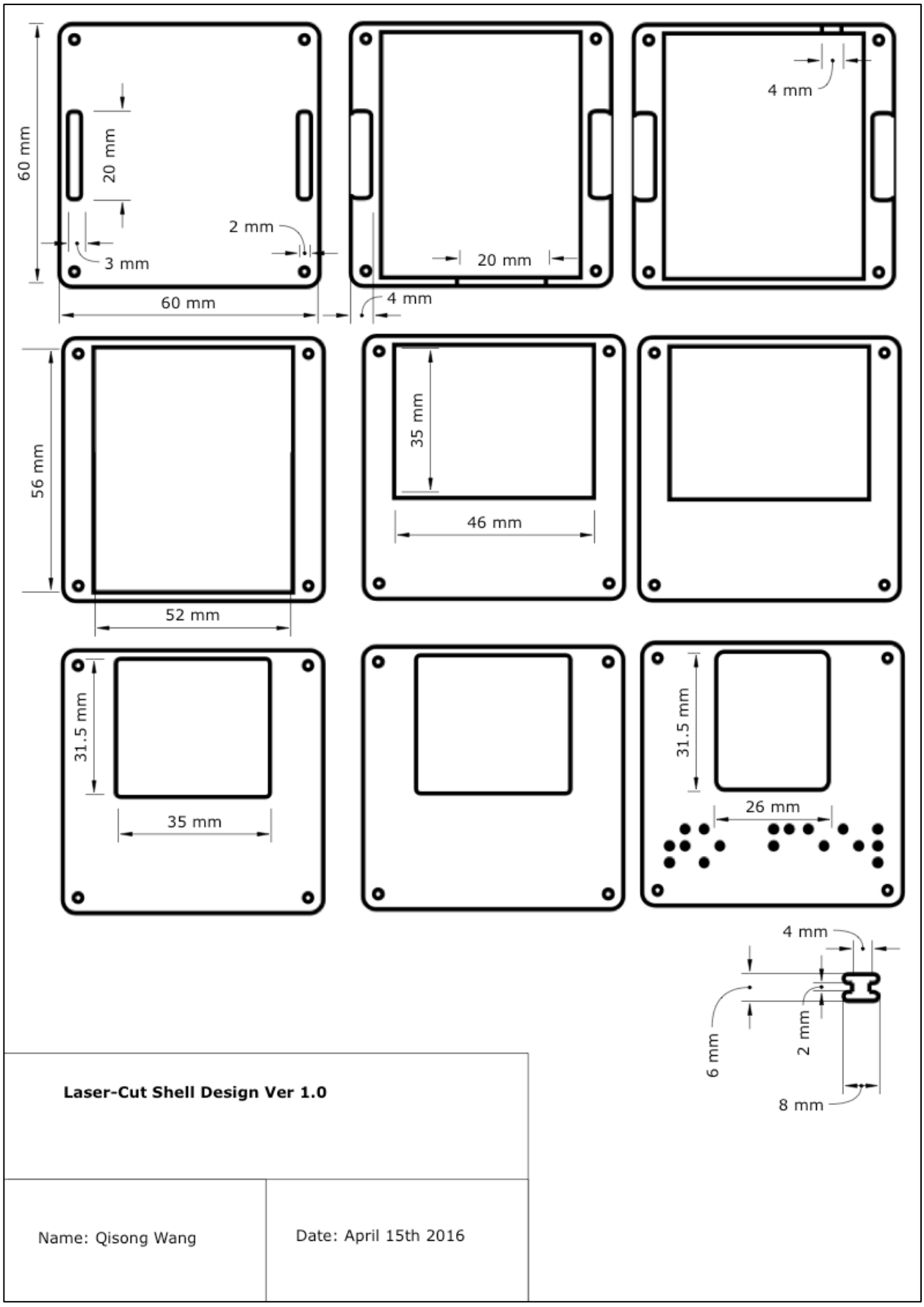


Figure 9-2: Shell Design 2D Drawing