Final Report

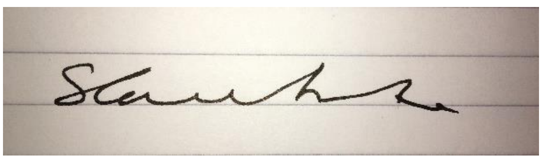
An investigation of Bluetooth as a communication protocol for wearable sensors.

Samuel Winchcombe

2019
3rd Year Individual Project

I certify that all material in this thesis that is not my own work has been identified and that no material has been included for which a degree has previously been conferred on me.

Signed



College of Engineering, Mathematics, and Physical Sciences
University of Exeter

Final Report

ECM3175/ECM3149

Title: An investigation of Bluetooth as a communication protocol for wearable sensors.

Date of submission: Friday, 03 May 2019

Student Name: Sam Winchcombe
Programme: BEng Electronic Engineering
Student number:650022548
Candidate number:009229

Supervisor: Dr Anna Baldycheva

Abstract

In recent years the Internet of Things (IoT), the use of the internet to connect 'things', has exploded and, with this, market wearable devices have emerged. The limiting factors on wearable devices are size, battery life, transmission rate and range. This paper aimed to assess Bluetooth as a suitable communication protocol for wearable devices by providing proof of concept and measuring peak current consumption and communication reliability.

Bluetooth's Low Energy (BLE) advertising mode was used to form a Wireless Personal Area Network (WPAN) enabling sensors to broadcast temperature readings to a master module. The prototype built consisted of two sensor modules broadcasting temperature values over ~1.5m to a master module. It was found that there was a trade-off between the advertising interval of the sensors and the current consumption which could be adjusted depending on the device's application. The lowest peak current consumption during transmission was recorded at 8.80 mA for an advertising interval of 7000ms and a transmission power of -23dBm. It was concluded that the approach outlined in this project was able to prove that modern versions of Bluetooth make a suitable communication protocol for wearable sensors.

With more time, three immediate steps have been identified to reduce power consumption of Bluetooth in wearable sensor applications:

- ***Use a newer version of BLE choosing from a selection of more customisable BLE modules to reduce the current consumed.***
- ***Adapt the code which is run by the chips to optimise the power usage.***
- ***Customise the advertising structure***

The effectiveness of these changes would be more accurately measured using average current consumption on an oscilloscope.

Keywords: Wearable sensors, WPAN, Bluetooth, Wireless Communication, I Beacon

Table of contents

1. Introduction and background

Wearable devices make up a large proportion of the growing Internet of things (IoT) market with an estimated 190.4 million units to be shipped in 2022, a Compound Annual Growth Rate (CAGR) of 11.6% over a 5-year period) [1]. The limiting factors for wearable devices are their size, battery life, transmission range and transmission rate, depending on their applications. As a result, most wearable devices are worn as smart watches as they offer an area large enough for a display, battery, processors, antenna (for relatively fast bit rate communication) and peripheral sensors. One of the reasons the battery needs to be large is the electric current required for communication. The most common Wireless Communication Protocols (WCP) are Bluetooth, Wifi and Cellular, and are often used in combination.

This project researched, designed and prototyped a Wireless Personal Area Network (WPAN) for a wearable graphene temperature sensor system. The proposed design had multiple sensors connected to a central node to form the WPAN using a WCP. The Central node sent data over a Low Power Wireless Area Network (LPWAN) to the cloud to allow real time data access from remote locations. The proposed network can be seen in figure 1 with a secondary connection method allowing sensor to connect straight to a smart phone. However, that is outside the scope of this project.

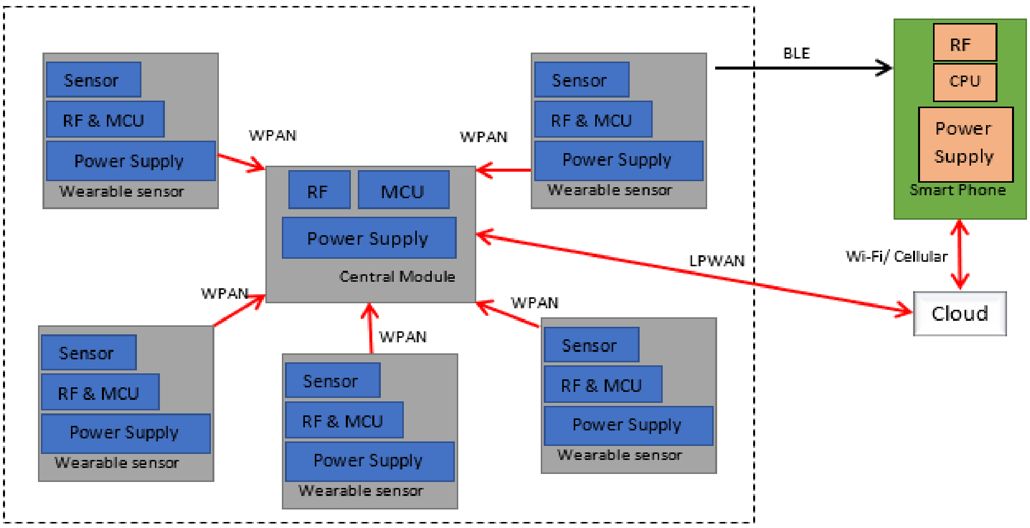


Figure 1: Proposed network design

1.1. WPAN’s used in wearable sensors

Recent developments in wireless communications for wearable sensing are being driven by their increased range of applications and as such use all the communication protocols in figure 2. Due to the scope of the project only WPAN’s were focused on here.

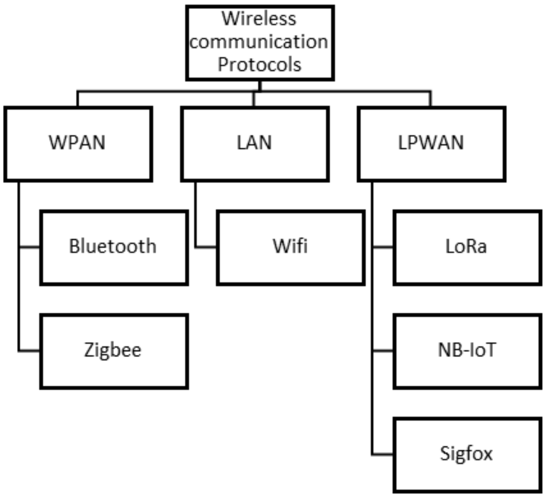


Figure 2: Categories of wireless communication protocols

WPAN

Bluetooth Low Energy (BLE) and ZigBee are the two of the most common WPAN protocols used in the IoT. BLE, as part of the recent Bluetooth 5 release, offers a low power communication network that transmits back to a device connected to the cloud (e.g. a smart phone). Operating in the 2.4GHz range of the Industrial, Scientific, Medical (ISM) band, BLE promises a battery life of up to 10 years with a Tx (transmission) power of 84 mW [2]. BLE also uses an adaptive data rate of 500 kbps, or 125 kbps when in the LE Coded connection type, to increase its indoor range to 40m [3], compared to its default 2Mbps data rate. Bluetooth can use a Star or mesh topology. Using these topologies, theoretically any number of end devices can be connected. The only constraints being the data throughput at the cloud connection of the host. An update in Bluetooth 5 increased the size of the advertising packets to 31 bytes [3] allowing larger data packets. Advertising packets allow the devices to communicate without synchronisation.

ZigBee, BLE’s main competitor in WPAN’s, operates in the 2.4GHz ISM band at: 250 kbps per channel, 915Mhz at 40 kbps per channel and 868MHz at 20 kbps per channel [4]. A 30m indoor range and, theoretically, unlimited range using its mesh network with unlimited devices. It comes at the cost of a slightly higher Tx power of 90 mW [2]. As the protocol is designed for the IoT it has many different standards that add specific network layers and application layers to the Physical and MAC layers which are defined in the IEEE 802.15.4 standard making it a versatile protocol.

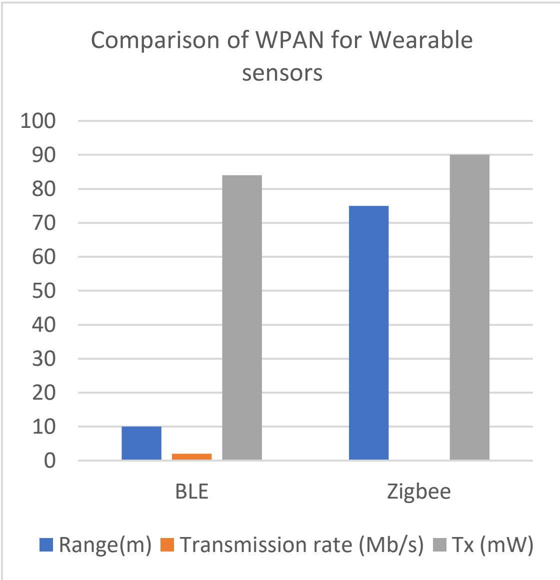


Figure 3:Comparing WPAN WCP

Comparing the two WPAN protocols with the limiting factors of wearable devices in figure 3, BLE appears to be the better protocol because of its lower transmission power and higher transmission rate. Although Zigbee offers a significantly longer range for the larger transmission power, the increased range is not needed in wearable sensor applications as the master module will never be that far away from the temperature modules. Therefore, the unnecessary range comes at the cost of higher power consumption. Bluetooth is also a well-established communication protocol which has been widely adopted by the world’s electronic industry, being installed in most smart phones. There are, therefore, a large selection of components that can be chosen from and a well-established support community in the form of forums, chip manufactures and the Bluetooth SIG (specialised Interest Group) available to provide technical support.

1.2. Back Ground of Bluetooth and SIG

Bluetooth was created by five companies in 1998 when they joined together to create Bluetooth SIG [5]. Named after a tenth-century King of Norway, Harald Bluetooth, for his unification of warring tribes, Bluetooth was intended as a wireless communication protocol to unite communications between personal devices [5] [6]. In 1999 the Bluetooth 1 specification was released royalty free to adopter members describing both its protocols and potential application scenarios and from 2000-2002 the first devices with built in Bluetooth were released[5][6]. In 2002 SIG entered Bluetooth version 1.1 to be part of the IEEE's 802.15.1 standard and it was selected as a baseline of the standard. [6].

In 2003 SIG adopted the core specification version 1.2 [5]. In 2004 core specification 2.0 was adopted by SIG to allow enhanced data rates to increase data rate transmission[5]. The first Bluetooth watch was registered with SIG in 2006, creating the first case where Bluetooth was used in a wearable device[5]. In 2009 SIG adopted core specification 3.0 HS allowing high speed communication and, later in the year, announcing the release of the core specification 4.0 which became available in 2010 which included BLE [5]. The aim of BLE was to introduce a new protocol stack for use in low power applications. In 2013 version 4.1 was rolled out adding new features to the protocol which allowed devices to support multiple roles simultaneously [5]. In 2014 and 2016 respectively, Bluetooth 4.2 and 5 were announced [5]. Version 4.2 was the introduction of IoT features and version 5 added more IoT features. In 2017 SIG added Bluetooth mesh networking to allow large scale device networking [5]. Currently celebrating their 20th Year anniversary, SIG has over 35,000-member companies and records over 3.6 billion Bluetooth product shipments annually[5].

1.3. Aim

The aim of this study is to investigate the feasibility of Bluetooth as a communication protocol for wearable temperature sensors.

1.4. Objectives

To achieve the aim several objectives will be needed to be investigated:

1. Understand the technical requirements needed for a communication protocol for wearable temperature sensors and compare the options available.
2. Research how Bluetooth is used today in wearable sensor applications in order to understand possible challenges for using it with a wearable temperature sensor.
3. Design the communication system and research component for compatibility and suitability against the performance requirements for a wearable device.
4. Test design and analyse the results to determine its effectiveness.

1.5. Format of Study

This study is split into sections starting with relevant background information, aims and objectives of the study. Section 1.1 addressed objective one and the literature review addresses the second objective.

The literature review explains the current state of the Bluetooth communication protocol used in wearable technology. Based on this, the project developed a theory to examine potential methods of forming the WPAN and a methodology to design a system prototype with its hardware and software requirements. This included exploring the processes needed to develop a working prototype. The project work involved a range of technical and practical challenges, some of which were unexpected. The project review section explores these and how each was overcome in order to achieve success. Finally, the presentation of results shows the finished project and results which were taken from it. This includes a discussion of the results, what has been learnt and how the project could be taken forward for future work.

The Project consideration section covers the process taken to keep the project on track and precautions taken to minimise risk to individuals and the projects timeline. Environmental and social-economic principles were also covered in this section. The report finishes with References and Appendix to cite information gathered for use in this project.

2. Literature review

Using Bluetooth as a Wearable Communication (WC) protocol for wearable devices involves assessing it against the same limiting factors as all wearable devices face, depending on their application - power consumption, range, data rate and size. As they are the most constrictive design parameters for wearable sensors, they will be focused on here to examine the past and current state of Bluetooth in current literature to analyse its strengths and weaknesses in each area. Current Bluetooth sensor networks will also be examined to identify how they have previously been used, including their strengths and weakness in these applications.

2.1. Analysis of the current state of Bluetooth for wearable applications

As wearable devices are battery operated, power consumption by all device components is important. In the specification for Bluetooth 1, three power levels were stated: 20dBm (100mW), 4dBm (2.5mW) and 0dBm (1mW) [6]. This allowed devices to adjust their power consumption and range depending upon their applications.

A wearable sensor system under development in 2006 for use by nurses used the low power consumption of Bluetooth to develop a communication system between wearable sensors and a main unit to allow the sensors to last over 8 hours on a 300mAh lithium-ion battery [7]. As the battery was also powering a Central Processing Unit (CPU) and accelerometer, this was an impressive battery life, but the scale of the wearable sensor was relatively large at 35mm (W) x 45mm (H) x 13.5mm (D) allowing for a larger battery and electrical components.

The most recent version of Bluetooth, Bluetooth 5 claims a power consumption of half of Bluetooth 4.2 in best case scenarios [8]. Another power transmission level was also added at 10dBm to allow more customisation of device energy usage. When compared to previous generations of Bluetooth and IEEE 802.15.4, Bluetooth 5's battery consumption was found to be lower in an indoor scenario although it was higher than Bluetooth 4.2 when tested outdoors with Line Of Sight (LOS) and no obstructions. This was due to the larger distance at which it was tested [8].

The reduced power consumption of Bluetooth 5 makes it well suited to power limited wearable sensors. Especially with an adjustable power usage allowing the energy used to be varied to account for range and transmission rates depending on the application.

Like power consumption, the effective range of Bluetooth has increased significantly over the generations. When Bluetooth 1.1 was released three theoretical distances were possible depending on the power consumption in operation. Bluetooth 1.1 could transmit over 1m, 10m, or 100m outside with LOS with no interference [9]. Bluetooth 5 is capable of unobstructed outdoor ranges of 200m and indoor range of 40m [9]. In more practical experiments Bluetooth 5 could transmit over 120m, roughly double Bluetooth 4.2 [8].

Wearable sensors, depending on their application and their network layout, do not necessarily require a large range as they generally transmit to a central internet connected device nearby. In the paper “Smart textile: Exploration of wireless sensing capabilities” by A. Somov, et al., it is shown that link quality does not decrease noticeably over a 10m range. Bluetooth’s option of ranges makes it especially suitable for wearable devices as the range can be adjusted to save energy. It could, however, waste energy if the device is transmitting further than necessary.

Bluetooth’s data rates have improved remarkably since the first generation of the standard. In Bluetooth 1.1 the maximum theoretical speed was 723 kb/s, and this came at relatively high power consumption [9]. By Bluetooth 5 speeds of up to 2Mb/s are capable at lower power consumption [8]. In applications, such as wearable devices, where the data rate from previous generations was already sufficient, the faster data rate still offers advantages. The faster a connection can be made; data transferred and disconnected the less energy will have to be used to complete the contact. Larger firmware updates to devices can also be completed quicker.

Lower data rates are available in Low Energy Encoded mode (LE). This increases the Signal-to-Noise ratio (SNR) and increases the range at speeds of 125kb/s and 500kbps. Whereas in Bluetooth packet headers and payloads are not encoded, in LE mode they are, with 1 bit referring to 2 modulated symbols at 500kb/s and 8 symbols at 125kb/s. This modulation can be used to save power by decreasing the bit rate and transmission distance.

Simple temperature sensors output requires very few bits, therefore, the minimal transmission rate can be utilised to save power in sensor modules.

2.2. *Current state of Bluetooth in wearable sensors*

In 2006 Ren Ohmura, et al were examining the use of Bluetooth Classic for wearable sensor networks. The network was formed using a piconet of B-Packs (accelerometer, battery, Bluetooth chip) where B packets were attached to the clothing of nurses that connected to a main unit also attached to nurses. The B-pack sensors measured the movement of nurses using the data from accelerometers as they carried out daily tasks. The design was prototyped and tested with Nurses at the TWMU (Tokyo Women’s Medical University) to examine the effectiveness of their network design and if it ergonomics. The device was capable of operating for over 8 hours, the duration of a nurse’s shift, before needing to be recharged. The nurses reported that when carrying out menial tasks they mostly forgot they were wearing them. It was only when helping patients they were noticed as the nurses were worried the sharp edges may cause harm.

Using a Piconet allowed the master module to hold up to 7 connections at once. This meant that 10 samples could be gathered before being transmitted without having to reconnect each time. Sending in intervals also saved power compared to transmitting continuously and reduced the interference from other sensor modules increasing the battery life of the B-packs. While in some scenarios 7 slaves may be a limiting factor, it was found to be adequate in this project as sensors were attached to limbs and the torso.

The network used in this experiment is effective but, compared to modern versions of Bluetooth, relatively power consumptive. Using a current version of Bluetooth, it would be beneficial to examine the effectiveness of this method. The scale of the devices could also be made smaller due to advances in chip design, materials and manufacturing method.

Providing an updated review of the connection requirements of Bluetooth, “Smart textile: Exploration of wireless sensing capabilities” by A. Somov, et al, provides an overview of both graphene in a temperature sensing application and an assessment of a Bluetooth communication system design. The manufacturing method of the graphene is explained, and its temperature sensitivity measured. The communication system was assessed in several scenarios using its Link Quality Indicator (LQI) and Received Signal Strength Indicator (RSSI). The graphene was able to achieve a resistance change of 80Ω/°C for the range of 25°C to 35°C. The temperature drop was measured using a temperature-resistance detector. The data from the detector was accessed from a smart phone using a BLE 4.1 connection operating at 2.4GHz.

The signal quality was tested using three scenarios over 10m: (i) Line of sight, (ii) in a box with 20mm holes and (iii) in a box with 2mm holes. The quality of the signal was measured using two parameters RSSI and LQI.

The results showed little degradation in RSSI over 10m and the obstruction of the box with different sized holes had minimal effect compared to that of the line of sight signal. It is not stated what material the box was made from. This makes it difficult to establish the penetrative strength of BLE 4.1. The LQI for all three scenarios was strong measuring above 200, however the signal which had a 2mm hole had an LQI which was 20 lower than the other two even for a similar RSSI. The 2mm hole’s signals LQI decreased by 10 over the 10m range compared to the other two which only dropped ~4.

This paper provided an up to date view of Bluetooth in a wearable sensor application compared with the work done by Ren Ohmura, et al. Investigating a one-to-one connection set up, the Ren Ohmura paper does not investigate the use of Bluetooth in a star network which this project did do. The paper does, however, show the effectiveness of a more modern standard over relevant ranges and shows its capability to deal with physical interference.

A 2017 paper [10] examined the two main modes of operation in BLE for an Opportunistic Sensor Data Collection (OSDC) scenario: connection based, and advertising based. The paper set the scenario of a fixed sensor transmitting data to a passing BLE enabled device by either advertising or a fixed connection. The paper examined the performance and trade-offs of BLE as a technology for OSDC. It analysed the theoretical and real current consumptions of both operational modes along with their maximum theoretical collectable data per contact interval. It used this to determine the collection interval in terms of energy consumed per collected bit. The effects of Bit Error Rates (BER) were considered throughout. The report finished by considering the limitations of hardware on the maximum collected data per connection interval.

When in advertising mode it was discovered that a longer interval between sending packets decreased power consumption and that only transmitting over one of the three advertising channels further reduced power but also decreased the reliability of receiving the packets. An advertising interval of over 3.06 seconds was found theoretically to allow a 230mAh coin cell to last over a year. As BER doesn’t affect the power of the transmission this had no effect on the battery life. When in connection mode the power consumption was much higher. At the largest transmit interval of 10.24s, the connection mode was found to use a factor of ~1 more energy than the connection mode. BER has an effect on the power usage of the connection mode however, as when a channel experiences a bit error the channel is shut down and the sensor sleeps until the next connection interval. As a result, when the BER increases the energy usage decreases as channels are closed more frequently. With these results it was calculated that when using three channels in advertising mode a sensor could last up to 3.02 years and a sensor in connection mode up to 1.09 years with a contact time (time devices are communicating) of 45 seconds [10].

Over a fixed period, the connection mode was found to be capable of transmitting more data irrespective of the advertising interval [10]. At intervals of 10.24s, advertising mode was found to be 4-5 magnitudes worse than connection mode [10]. It was also found that when transmitting data up to 334kbits per contact, advertising mode was sufficient but if higher data rates were required connection mode could manage 10Mbits to 100Mbits depending on the duration of the contact [10]. BER was found dramatically to reduce this but it is still a lot better than advertising mode [10].

To investigate the real-world limiting factors of transmission distance and the parameters of the connection interval, two Bluegiga BLE121LR platforms were set into master and slave modes. The maximum measured throughput per contact interval was 156.5kbit/s. This was higher than found in other literature or that reported by the manufacturer but still lower than the theoretical maximum, as would be expected. This was possible due to the timer-based application that was used for generating and sending data. The throughput was found to be limited by the maximum number of round trip exchanges per connection interval allowed, which was found to be 11. These results were no surprise as BLE is not optimised for high throughput applications [10].

To test the effects of distance on throughput, two scenarios were tested, on a university campus and on a beach. The throughput was found to be constant up to ~200m in both scenarios after which the throughput decreased in both cases. In the university set up however, there were oscillations in throughput starting at 300m as the characteristics of multi-path propagation from the surrounding buildings were experienced.

There are multiple reports examining the applications of Bluetooth in remote sensing scenarios using both Piconets and advertising mode. They also measure the effectiveness of Bluetooth in these applications in terms of link quality and power consumption. However, no recent experiments have been found to test the feasibility of using Bluetooth to connect sensors in wearable sensor applications using BLE 4.0 or more recent versions of Bluetooth. Measuring the effectiveness of the design in this project will require use of the same matrix's as previous research as the results are comparable. Deciding whether to use a piconet or advertising to form the network will depend on the requirements of the system and the required throughput from the sensors but the literature reviewed proves that both methods are possible.

3. Theory and Methodology

The focus of this project is testing the suitability of the latest Bluetooth protocol as WCP in wearable sensor application to connect multiple temperature sensors to a central module and as such this section will cover the theorical approaches and the planned methodology that was used to achieve it. The planned test used to analysis transmission quality and power consumption will also be used here.

3.1. Theory

Bluetooth offers several ways to connect multiple devices allowing communication, one of which is a piconet. A piconet is a collection of devices configured in a star network formed in an ad hoc manner. As seen in figure 4 it consists of a master and up to 7 slaves. The master device can be any device as their roles are not specified at manufacture. Masters control the network deciding who transmits and when. While a piconet can only have up to 7 active slaves it's possible for the master to register the addresses of 'parked' slave devices in the area allowing it to disconnect an active slave and transferring its active member address (AM_ADDR) to the parked slave adding it to its network [6].

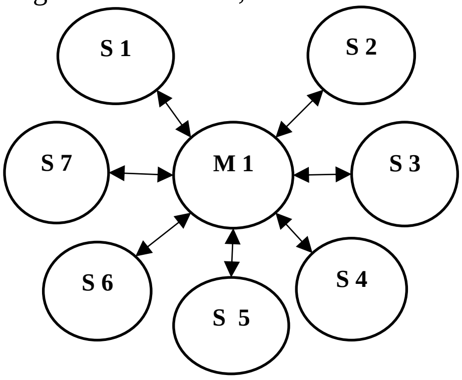


Figure 4 Piconet Diagram

The communication channel of a piconets is made of a sequence of frequency hops. Each transmission slot is 625µs. If a transmission extends over 625µs multi-slot transmissions are possible [6]. In this case the frequency of the transmission is kept constant for the duration of the slots before the frequency hopping resumes as it would for a single slot message. Slots are identified as odd or even depending on the value of the second least significant bit. Master and slave alternate transmission slots using Time Division Duplexing (TDD) with the master transmitting on even number and slaves on odd [6]. Slaves can only transmit after receiving transmissions from their master. Timing synchronisation is maintained by the slave using the master's Bluetooth clock, the duration of slots and the offset of its own clock to the Masters [6].

For BLE Carles Gomex, et al theoretically calculated that hundreds of slaves could be connected to a master depending on the connection interval parameters [11]. However, it is noted that practical limitations such as memory and antenna availability would limit this number dramatically [11] .

The second method is to use advertising, as seen in figure 5. Advertising packets were added in 4.0 BLE specification to allow a device to broadcast their data to nearby Bluetooth devices. This allow unidirectional communication without connection or synchronisation. If the receiving device wants further information from the transmitter then a connection can be formed.

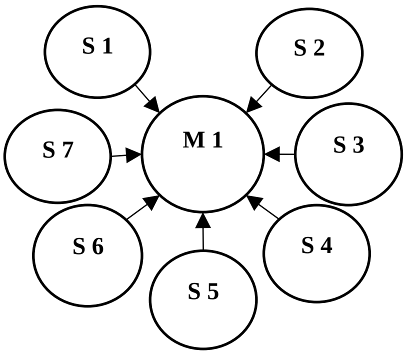


Figure 5 BLE Advertising Mode

Operating in the 2.4 GHz spectrum between 2402MHz – 2480MHz BLE's advertising channels operate on three of the 40 1MHz channels. Although BLE channels are separated by 1 MHz, the three advertising channels, 37, 38, 39, as seen in figure 6, prevent the chance of multiple channels being blocked by the same disturbance. This is unlikely as most interference comes from narrow band sources, for example Wi-Fi, Bluetooth classic, microwaves and baby monitors [12]. Channel 38 is carefully positioned between the Wifi channels 1 and 6 to avoid interference.

Advertising intervals can be varied between 20ms and 10.24ms in steps of 0.625ms. A random delay pseudo time of 0-10ms is added to the fixed interval to reduce the possibility of collision with other

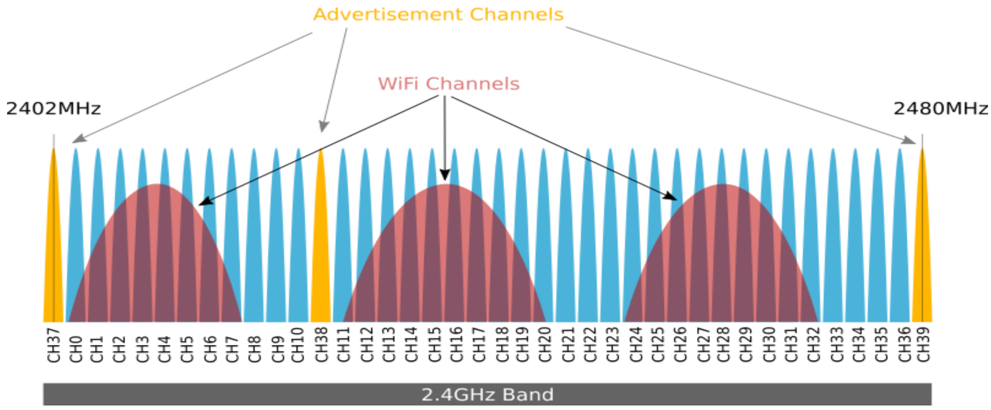


Figure 6: Frequency spectrum of BLE [12]

advertising packets. The longer the interval durations are, the lower the energy consumption is.

The top-level packet, defined in the Bluetooth specification for BLE, is declared as two data units. The packet itself has several parts as seen in figure 7. The PDU (Packet Data Unit) for the advertising channel includes a 2-byte header and a variable payload from 6 to 37 bytes defined by the 6-bit Length field in the header of the advertising channel PDU.

There are two main types of PDU:

1. ADV_IND: for generic advertisement and it is the most commonly used. Non-direct and connectable modes are possible. It helps devices to be found by master devices to allow connections.
2. ADV_NONCONN_IND: used in beacons when connections are not wanted.

An Advertising PDU has a payload depending on its advertising PDU type. The payload contains a 6-byte advertisement address (MAC address) and a variable payload size of up to 31 bytes, which is made up of a custom advertisement data structure. The advertising data types, which are combined to make an advertising data structure, are defined on the SIG website. Each type specifies a different standard of data in the payload. These data types can be combined to suit the application. For example, Apple

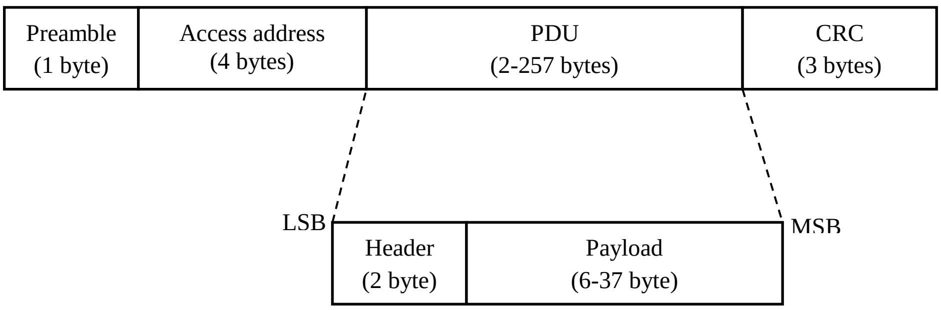


Figure 7: Data structure of advertisement packets [13]

created its own data structure for I Beacons to combine standard advertising data types with manufacturing specific ones [12].

Custom Universal Unique IDentification (UUID) at the beginning of the advertisement data structure are used in many data structures to identify what the advertiser purpose is and to distinguish it from other advertisers so that master device doesn't have to connect to decide if it is the beacon that it wants to connect to. UUID's are made up of 128 bits and take up a large portion of an advertisement packet. There are specified ways to define UUID's in form using only 16 bits to group use purposes together, for example '0x180D' is the 16-bit UUID for a heart rate service [12]. In Bluetooth 5 the advertising packet size was increased from 31 bytes to 255 bytes to allow more data to be included in the transmission [12].

3.2. Methodology

Having looked at the two BLE connection modes used in wearable sensors, Bluetooth's Piconet mode was selected due to its long standing use and the bi-directional communication it allows between multiple slave nodes and a master node. As it is unlikely that someone will be wearing more than 7 sensors, the limitation should not be a problem.

The network will be made up of two types of modules: sensor modules and a master module. The sensor modules are responsible for gathering data from their surroundings, in this case temperature data, and transmitting it to the master module. The master module is responsible for maintaining communications with slaves and storing the data received from the sensor modules for further transmission over LoRa a LPWAN.

The software on the master module will follow the flow diagram in figure 8 while the sensor modules will aim to follow the flow diagram in figure 9. While the Master module has not taken full advantage of its bi-directional communication, as its outside the time period of this project, it does allow further expansion by allowing the master module to edit the setting of the sensor modules.

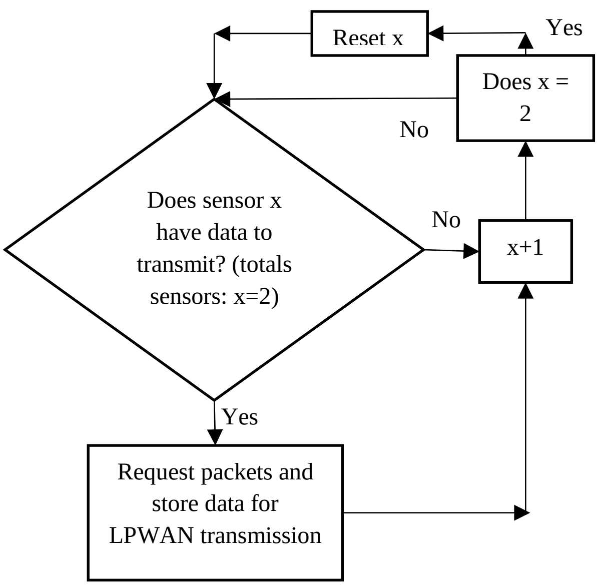


Figure 8: Flow diagram of master module

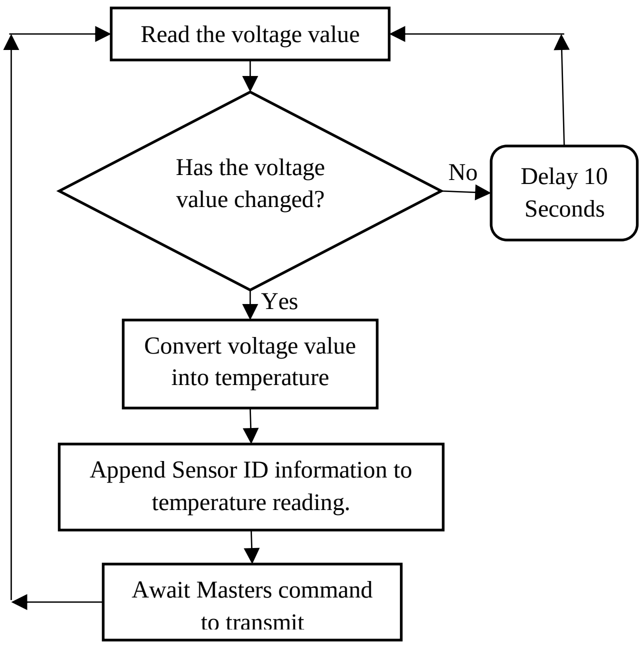


Figure 9: Flow diagram of sensor modules

3.3. Hardware Requirements Selection

The Sensors modules and Central module will communicate over BLE, which has the advantage of being a popular wireless protocol, so chips are widely available. The Sensor modules require a temperature sensor to simulate the signal output from a graphene temperature sensor. Both the Sensor module and Central module require a Microcontroller Unit (MCU) to control the BLE module and process data from the sensor. The BLE module selected to prototype the slaves was a Microchip RN4871 BLE module.

The RN4871 has a 16-bit ADC, BLE 4.2 chip, with integrated MCU for hostess applications, integrated antenna and a compact form factor [14] [15] . Measuring only 9 x 11.5 x 2.1 mm, the RN4871 is compact and its operational voltage range of 1.9-3.6V makes it ideal for wearable applications as it operates at nearly half of the supply voltage range from a 3V coin cell battery. Due to the project's budget, this chip could not be bought with a breakout board. As a result, there was a risk that the chip could be damaged by static or heat during the soldering process due to the pins proximity to each other and the limited soldering equipment (no electronic microscope with thin tipped soldering irons) available at the University of Exeter. The RN4871 also came with easy to use AT commands for configuration.

As a backup a BLE 4.0 HM-10 modules could work as a proof of concept. As BLE 4.0 is the first generation of BLE the module has fewer features than the RN4871. It also operates from 2V to 3.7V and so would operate within the voltage range of a coin cell battery. Based on Texas Instruments' BLE 4.0 CC2540 chip, the chip is in its third iteration and so would be reliable due to its many design iterations and popular consumer adoption [16][17]. To edit the firmware, however, would require the TI Compiler, IAR 8051 IDE, which was well beyond the budget of this project. This meant the HM-10's firmware could not be edited reducing the customisability of the module. The 12bit ADC of the CC2540 is not given a pin on the HM10 but could be accessed by soldering a wire to it if necessary. An active mode current of 8.5mA but a sleep mode current of 400µA~1.5mA and an adjustable RF power allow for remote sensing applications [18].

For the Master module a RM186-SM-01 BLE 4.1/LoRa module from Laird was selected because of its form factor, integrated antenna and being one of the only modules on the market which has a WPAN and a LPWAN on one module. Capable of operating from 1.8V to 3.6V it is suitable for wearable devices and '*smartBASIC'* commands (AT equivalent commands) which makes it easy to programme [19].

An Arduino Uno is an ATmega328P microcontroller unit from Microchip, built on a breakout board with 14 digital I/O pins (6 of which are PWM pins), 6 analogue I/O pins, a 3.3V and 5V supply pin [20]. The Arduino Uno was planned to be used to program the RN4871's integrated MCU and, if needed, to run the HM-10 modules. It is programmed through a USB connection to a computer and paired with an Integrated Development Environment (IDE) for script development. The University has a number of these available for use and so did not come out of the projects budget.

To simulate the output from a graphene temperature sensor a TMP 36 was used. Accurate to +/- 1°C at 25°C and with a linear relationship between output voltage and temperature increase. A 10mV increase is equivalent to an increase of 1°C [21]. This is not as accurate as the 80Ω/°C recorded from the graphene temperature sensor in the 'Smart Textile: Exploration of Wireless Sensing Capabilities' paper. However, analysing whether the ADC of the Bluetooth are accurate enough to take account for a graphene sensor is outside the scope of this project.

3.4. *Software Requirements*

It was decided the best suited programming language was C++. It is compatible with the BLE chips selected and the author has personal experience with C# which was sufficiently similar to allow coding and debugging to be done quickly.

The software requirements for this project were:

a) Reading voltage readings from the sensor
b) Converting mV reading to temperature value
c) Transmitting temperature and senor ID data.
d) Passing data to LPWAN.

The above requirements had to be accomplished with minimal current consumption compared with the current consumption in the “Opportunistic Sensor Data Collection with Bluetooth Low Energy” article [10]. The sequence proposed to achieve these requirements can be seen in figures 8 and 9.

4. Design Process

After confirming chip feasibility and getting purchasing approval from the project’s supervisor, Dr Anna Baldycheva, the chips were purchased. Initial plans to create a breakout board for the RN4871 were curtailed by worries from Peter Armitage, laboratory technician, that the University of Exeter did not have the facilities to create a breakout board with traces small enough to accommodate the RN4871’s footprint without short circuiting the pins. To first confirm the projects concept, it was decided to prove that a connection could be made to transmit the data from a sensor over BLE. This was achieved using a HM-10 and an Arduino Uno as the MCU to transmit the temperature from a TMP-36. The set up and transmission, which was picked up from an I Phone 5S using a Bluetooth Serial App, can be seen in figure 10. The temperature was calculated by dividing the 5V range of the ADC by the 1024 levels (2^10). Subtracting a 0.5V offset and converting the V into mV before dividing by the 10mV linear relationship.

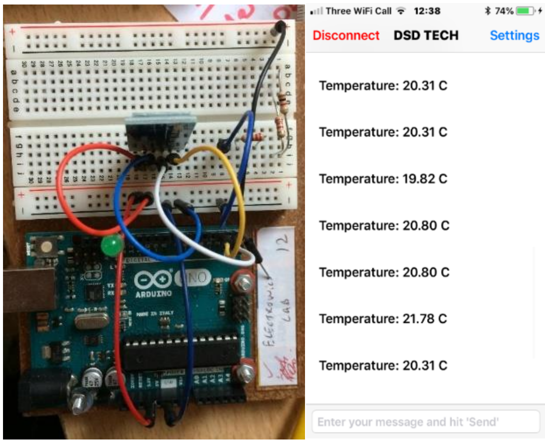


Figure 10: Bluetooth setup and serial communication output

After establishing this connection, and having researched further into piconets, multiple documents described how specific stack layers are required in devices to create a piconet, which neither of the modules purchased had.

To overcome this problem three Firebeetle Board-328P with BLE 4.1 were purchased, outside of the project budget, due to their ability to form a star network allowing the proposed network topology to be created [22]. The module’s communication with the Arduino IDE seemed to be intermittent. However, there was limited information on the datasheet and, with little data online from which to draw conclusions about how to proceed, progress quickly slowed other than putting the modules into star mode. Posts on RDF Robotics forums also resulted in nothing. With limited time left and limited support available online or from University technical staff to make the Firebeetle boards work, it was decided to investigate BLE's advertising mode as an alternative solution to forming a Piconet.

Talking to Roger Perret, R&T (Research and Technology) Unit at the University of Exeter, it was understood that is would be possible to manufacture a breakout board for the RN4871 within the University of Exeter. The circuit diagram made for the breakout board can be seen in figure 11 and was made in Multisim. The breakout board itself was designed in Ultiboard and can also be seen in figure 11. The RN471 was not a stock component in the National Instruments Library and as such a layout-only component was constructed using the component wizard tool in both Ultiboard and Multisim, as seen in figure 11. With no similar templates on which to base the module, the footprint had to be custom made following the information on the data sheet for dimensions of its foot print and location of the keep out areas to prevent interference with the antenna [23]. The recommended reference circuit was used to design the rest of the breakout board although the configuration interface was not followed exactly as matching connectors could not be sourced. Therefore, pin configurations on the interface were less important to follow as wires would be soldered to the board to allow flexibility.

A footprint for the STS2301 power transistor also had to be created as the component was not in the built-in library. The STS2301 had to be ordered to protect the circuit from reverse voltage spikes or incorrect wiring of the Vcc and Ground, and the University had none in stock. A pull up resistor circuit was attached to the $\overline{RST}$ pin to stop the logic level fluctuating on that pin.

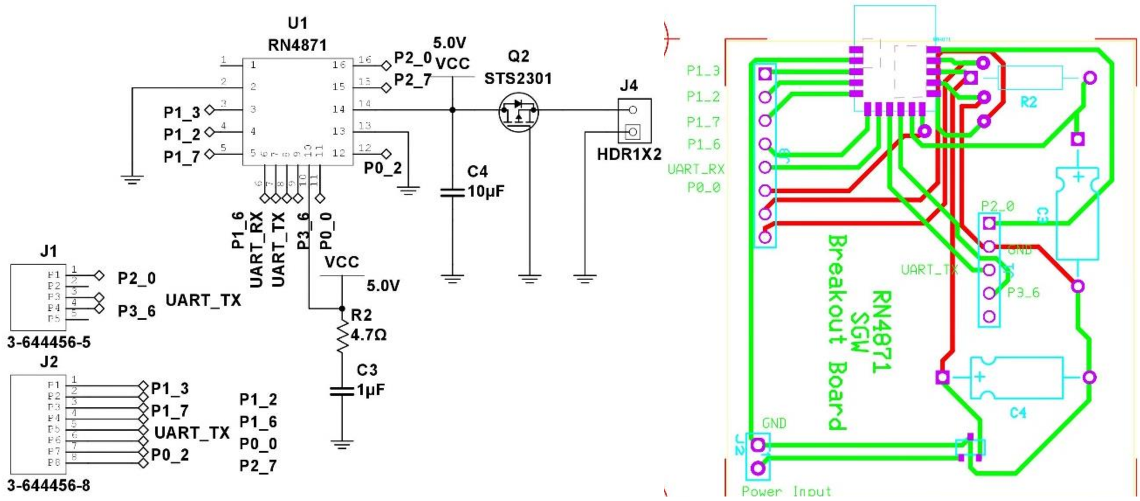


Figure 11: (left) Circuit diagram of Breakout board, (Right) PCB design.

Having been checked by Roger Perret and printed, the breakout board was assembled with the RN4871 extending over the edge to ensure the least interference with the antenna. The module was soldered to the board by Peter Armitage, taking full advantage of his superior soldering experience. When the board was wired to the Arduino however, there was no activity. Using a multi-meter, it was determined that the correct voltage was entering all the pins but scans for the device from multiple Bluetooth scanning application did not reveal the Bluetooth chip The STS2301 was replaced in case it had been damaged by static or the soldering process, but this had no effect. The STS2301 was then replaced with a reverse diode across the Vcc and ground pins to protect against reverse wiring and reverse voltage spikes, as seen in figure 12. This also had no effect. A final check to see if the RN4871 was dead was made. An LED was put in series with an 330Ω

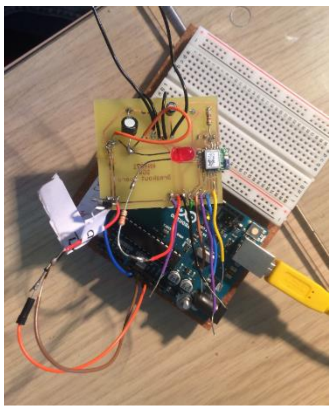


Figure 12: breakout board wired to Arduino

resistor and connected to P0_2 to test for power within the module. This showed there was none. It was, therefore, hypothesised that the module had been damaged at some point by either static charge or the heat of soldering the module to the board. The final state of the board can be seen in figure 12.

If future prototyping is possible with a larger budget, the RN471 should be purchased with its breakout board to allow rapid prototyping and reduce the chance of it being damaged.

With the remaining budget from the project a second HM-10 module was bought, and a second Arduino Uno acquired from Peter Armitage. Later a third HM-10 and Arduino were purchased outside of the project budget to provide full proof of concept. Using HM-10’s earlier had been avoided due to Bluetooth 4.0 going through depreciation and, therefore, no longer being supported from July 2020 by Bluetooth SIG [24]. A new Bluetooth device using 4.0 cannot be registered with Bluetooth SIG. The HM-10 would still show proof of concept although with fewer features and smaller advertising packets than later version of Bluetooth.

The HM-10’s were connected to Arduino Uno’s to act as MCU’s and TMP-36’s were again used as the temperature sensor on the sensor modules as seen in figure 13. An external library was added to the Arduino IDE to allow soft Serial ports to be used. The ‘AltSoftSerial’ library allowed digital pins to emulate Serial ports from the Arduino module [25]. This allowed the main serial command to be used to display outputs adding more versatility to the MCU. The output from them, however, is 5V and as the Rx pin on the HM-10 is only 3.3V tolerant meaning the voltage had to be reduced. The output from the Arduino was, therefore, put through a potential divider circuit. The Tx pin is 5V tolerant so did not have to be adjusted.

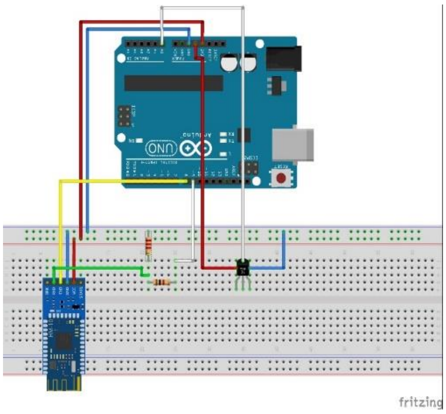


Figure 13: Sensor module set up

The HM-10 has the built-in advertisement structure to be used as an I beacon and, therefore, the advertising packets are limited to using their data structure and the data types they contain. This means the full 31 bytes are used in the payload even though not all of these are needed as the I beacon is designed to be used to find the advertisers location (for example, a shop) rather deliver a sensor relevant data.

The master module set up as shown in figure 15 executes the sequence in figure 14:

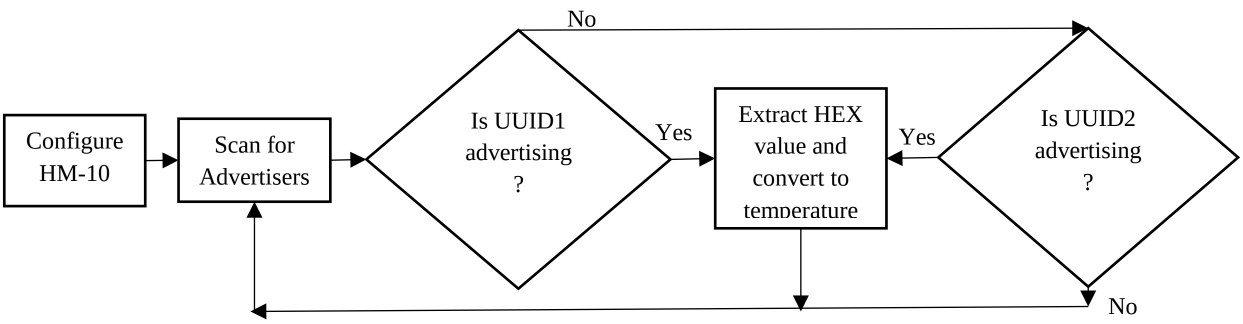


Figure 14: Flow Diagram of Master Module

The I Beacon advertisement packet from the HM-10 Module using BLE 4.0 has the following format:AAAAAAAA:BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB:CCCCDDDDEE:FFFFFF FFFFFF:GGG

A = Company ID

B = UUID

C = Major Field

D = Minor Field

E = RSSI (from Device)

F = MAC Address

G = RSSI (dBm)

The UUID field is used to identify which Sensor module the packet has come from and the Minor field is used to transmit the temperature reading. The Master Module needs to be capable of picking the sensors' packets out of all the advertising packets currently being transmitted and, therefore, looks for the UUID's of the Sensor modules. If these are found the Mino field is extracted using a sub-string between the two-colon which are located by searching for their ascii character from the end of the UUID. Their values mark the start and end of the sub-string. The Hex value can then be extracted by making a second sub-string of the Minor field before being converted to an unsigned long data type and being converted into a temperature and stored for that sensor.

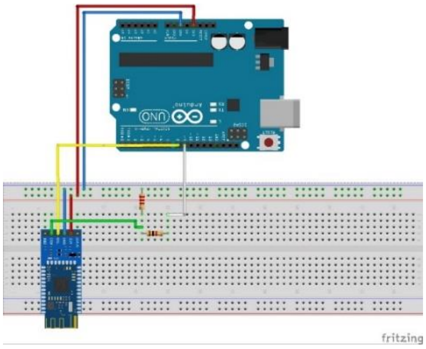


Figure 15: Master Module set up configuration

The code at the Sensor modules aimed to execute the sequence in figure 16:

A lot of time was spent creating this sequence as there were several problems to overcome. The first problem encountered in this process was adding the initial AT set up commands to configure the advertisement mode as the response from the HM-10 had to be checked against what was expected to be received to ensure no error had occurred. This worked by comparing the response from the HM-10 against an expected response. If they did not match, then '1' was added to a counter and the counter was printed after each command to allow quick debugging of commands.

The second problem encountered in this process was adding a variable to the AT command that changed the Mino field, as the temperature reading was continuously changing. The AT command only accepted an ascii string and as such could not just take a variable. To accomplish this the Mino AT command was made up of two pieces. The first section 'AT+MINO0x00' was stored as a string and the temperature, once converted to hex, was converted to a string and amended to the end of the first piece creating the full Mino field AT command. This was then written to the Soft Serial ports to be sent to the Hm-10. This method limited the voltage readings to 255mV (FF) as the first 2 values of the string were fixed. However, this wasn't a problem as 255mv resembles 74.51˚C. Outside of the range of temperature this project was interested in.

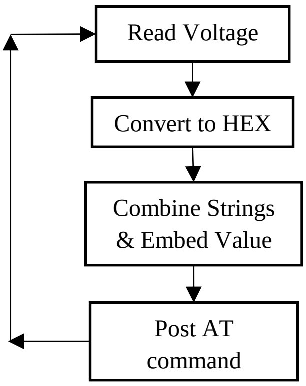


Figure16: Flow diagram of Slave module

With the code now working it was realised that the module was losing the accuracy of the temperature sensor as the floating-point temperature value was having to be converting to an integer to then be converted to a HEX value. To avoid this the code was adjusted so that the raw data (millivolt reading) was sent, rather than the temperature, as this was already an integer value and the conversion to temperature would be done at the Master module.

Up until this point only HEX values with numbers in had been transmitted due to the sensor being at the same temperature as the room. When the Sensor module was placed under a lamp however, it was noticed that HEX values, which contained letters, were not being sent. After further examination it was realised that the AT commands had to be uppercase letters as the commands were being sent as ASCII characters rather than actual HEX value and, in this format, their binary values were not the same when being read by the HM-10. A function was, therefore, introduced which searched for letters in the HEX string and converted then to capital versions of themselves before returning the string to be appended to the end of full Mino Command.

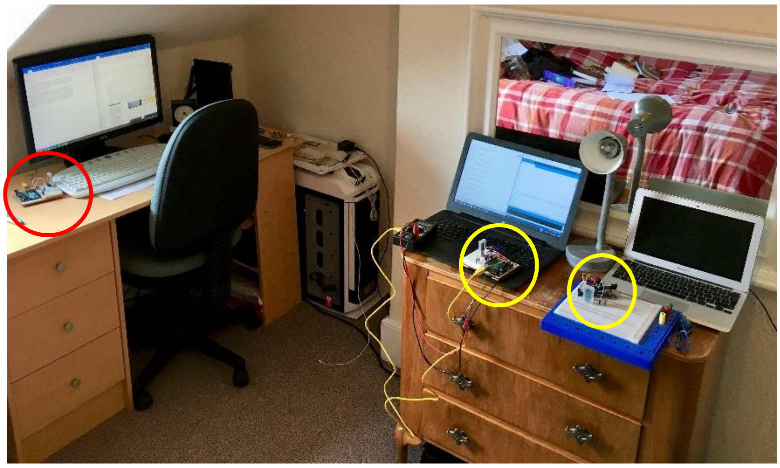


Figure 17: Experimental set up of Master module (circled in Red) and the sensor modules (circled in yellow)

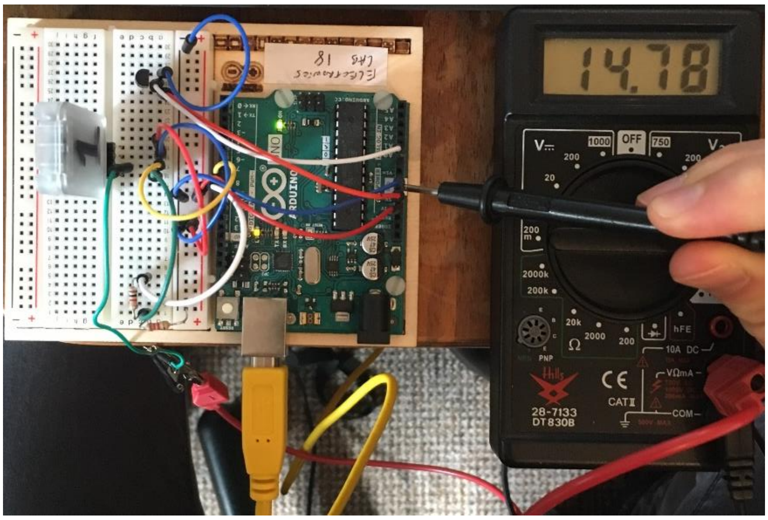


Figure 18: measuring TX current consumption

Setting up a second sensor uncovered an oversight in the initial AT commands as the reset command was overwriting the UUID’s of each HM-10 making them all the same (default). To fix this an AT command was added to the initial set up to set the first 8 bytes of each UUID for the module. The final proof of concept experiment and readings were carried out with two sensor modules and a master module separated by a distance of ~1.5m as shown in figure 17. The distance was chosen because it represented the upper limits the wearable sensors would have to transmit over. Computers were used to power the modules and display their outputs. Initially the sensors were set to the longest transmission interval and the default transmission power setting (0dbm). The experiment aimed to understand two things: the factors which effect the reliability of transmissions over ~1.5m and measuring the current consumption under different transmission parameters (TX) in the BLE HM-10 module. The transmitting current was recorded, using a multimeter as seen in figure 18 and the transmission characteristics were changed using the AT commands in the sensor’s scripts. The reliability was measured by counting the number of times each sensor appeared in 20 scans. This was repeated 3 times and the average number of times each sensor appeared calculated. The average of the two sensors was then taken and divided by twenty to produce the probability of a sensor being detected in each scan. The equation can be referenced as equation (1). Squaring this probability gave the probability of both sensors appearing in a scan. Between each set of 20 the sketches were reuploaded to the Arduino to ensure the timings of the transmissions and scans were random.

Control variables were kept constant during the experiment to prevent uncontrolled effects on the result, for example the Modules separation distance was kept constant and the USB port powering the Arduino was not changed during the experiment in case its outputs was different from the others.

Once the experiment had started the modules were not moved. This kept the distance between the modules constant and prevented it from affecting other variables.

$$reliability = \frac{\left(\left(\overline{No^{\circ}\ of\ scans\ sensor\ 1\ appeared\ in} + \overline{No^{\circ}\ of\ scans\ sensor\ 2\ appeared\ in} \right) / 2 \right)}{20} \tag{1}$$

The current draw was measured on the TX pin of the HM-10 under each change in transmission parameter to examine the effects the parameters had on the efficiency of the module.

5. Presentation of prototype results

The work from this project has provided proof of concept with a working prototype. The experimental set up can be seen in figure 17. The prototype successfully connects two temperature sensors to a master via BLE using its advertising mode. The Sensor modules and Master module executed the flow diagrams shown in figure 19 and 14 respectively.

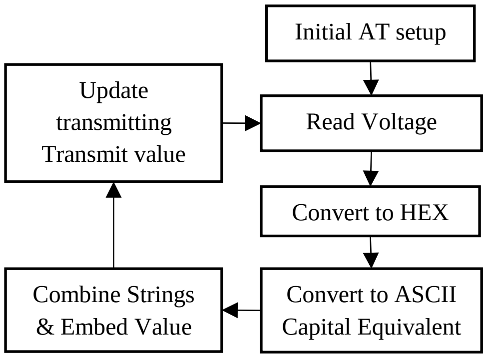


Figure 19: Flow diagram for Sensor modules

5.1. Results

The fourth objective of the paper was to create a working prototype that used BLE to send data from multiple sensors to a Master module and to measure its current consumption to provide an indication of a device's theoretical battery life. The first of these was achieved as the temperature reading taken from the two sensors, see figures 20, were transmitted to a Master module as seen in figure 21. The device succeeded in maintain the accuracy of the TMP-36 with temperature measurements being recorded to 0.01 degrees Celsius. The difference in measurements between the two sensors was primarily due to the TMP-36 chips not being identical, a result of being mass produced. In addition, the Arduino's were not supplying exactly 5V and, as a result, the outputs from the TMP-36's did not match the theoretical expectation. However, it should be noted that they behaved properly and showed an increase in temperature when placed under a lamp. Due to the advertising intervals being set to 7000ms not every transmission was picked up by the Master module. This can be seen in figure 21 where, in the second string of advertisers discovered, only sensor one is present.

```
20:08:18.304 -> AT+MINO0x008D->OK+Set:0x008D
20:08:19.794 -> ADC reading:141
```

```
20:08:16.572 -> ADC reading:163
20:08:17.073 -> AT+MINO0x00A3->OK+Set:0x00A3
20:08:18.569 -> ADC reading:162
20:08:19.073 -> AT+MINO0x00A2->OK+Set:0x00A2

20:08:20.550->ADC reading:162
20:08:21.052 -> AT+MINO0x00A2->OK+Set:0x00A2
20:08:22.549 -> ADC reading:163
20:08:23.048 -> AT+MINO0x00A3->OK+Set:0x00A3
```

Figure 20: Examples of transmissions from (top) sensor 1 and (bottom) sensor 2

The results from the experiment described at the end of section 4.1 are shown in table 1. From the results it can be seen that the Transmission power didn't have an effect on the reliability of the packets as the difference in probability between the two power levels for the same transmission rate were negligible and would have reduced further with more samples. This was because over 1.5m with no obstruction, the decrease of transmission power did not stop the sensors' data reaching the Master module. The transmission rate, however, did have a significant effect reducing the chance of a module being detected by up to 67% and 81% for both modules being detected by the same scan. This was due to the transmission rate making it less likely for the Master module to be scanning when the sensors were transmitting.

```
20:08:18.577->OK+DISISOK+DISC:00000000:00000000000000000000000000000000:0000000000:00989A113F4D:-069
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:11EBD9F4B2D8:-066
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:6ACA3DA98E6F:-092
              OK+DISC:4C000215:74278BDAB64445208F0C720EAF059935:123400A2C5:D8A98BB13644:-074
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:489F46CE8370:-058
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:7DFBC4799A75:-080
              OK+DISC:4C000215:12345678B64445208F0C720EAF059935:1234008DC5:D8A98BB0D073:-091
              OK+DISCE
20:08:19.140 -> Sensor 1:
20:08:19.206 -> Temperature: 29.10 Degrees Celsius
20:08:19.272 -> Sensor1 found
20:08:19.272 -> Sensor 2:
20:08:19.340 -> Temperature: 18.85 Degrees Celsius
20:08:19.373 -> Sensor2 found
20:08:23.348->OK+DISISOK+DISC:00000000:00000000000000000000000000000000:0000000000:11EBD9F4B2D8:063
              OK+DISC:4C000215:74278BDAB64445208F0C720EAF059935:123400A2C5:D8A98BB13644:081
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:00989A113F4D:071
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:7DFBC4799A75:087
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:6ACA3DA98E6F:091
              OK+DISC:00000000:00000000000000000000000000000000:0000000000:489F46CE8370:-066
              OK+DISCE
20:08:23.849 -> Sensor1:
20:08:23.915 -> Temperature: 29.10 Degrees Celsius
20:08:23.948 -> Sensor1 found
20:08:23.948 -> Sensor2 not found
```

Figure 21: Master modules received advertising packets from sensors and its output as temperature values.

Table 1: Results from prototype experiment

Transmission Power (dBm)	Transmission Rate (ms)	Current Consumption (mA)	Probability of a sensor being detected	Probability of both sensors being detected
0	7000	11.68	0.38	0.15
0	100	14.78	1.00	1.00
-23	7000	8.80	0.33	0.11
-23	100	12.17	1.00	1.00

Changes in transmission rate and power effected the current consumption of the Sensor modules. Changes in transmission power reduced the current consumption by 2.75mAh on average which, when operating of a 1000mAh, is a significant saving. Reducing the transmission rate also reduced the current consumption by an average of 3.21mAh, again, a big saving. The Vcc pin of the HM-10 drew a lot of current, 0.12 Amps, which, for a device intended for wearable applications, is very high. However, as this was a relatively cheap chip and designed for remote sensing applications rather than wearable sensing, this was not included in the decision of BLE’s suitability as a WC protocol. There are also simple adjustments to the script to save power, for example turning the advertising packets off between transmissions and putting the HM-10 into sleep mode reducing its current consumption to µA between temperature readings. These two changes would significantly reduce the current consumption of the

Hm-10 module but were not done in this project as proof of concept has already been achieved and the low power aspect of the code already witnessed. With these changes, it is believed current usage would be acceptable for a wearable device.

The probability of a sensor being detected has been measured accurately using replicates of measurements to reduce uncertainty due to random errors allowing outliers and anomalies to be spotted. Testing transmission rates between 7000ms and 100ms, with 5-7 replicates per transmission rate to further reduce uncertainty, would allow better analysis of the trade-offs between reliability and current consumption. It was not deemed necessary in this project as a goal was to show proof of concept and analysis the effect the programmable parameters have on the current consumption. The current consumptions represent physical values but does not display the current consumption as a function of time. They, therefore, do not show the average current consumption and its fluctuations in current consumption as the sensors transmit. With further research such a function could be built into the Arduino and automated to remove human biases and errors in reading values. An alternative, and potentially more accurate method, would be to use an oscilloscope to measure the voltage across a 10Ω resistor added at the TX pin. 10Ω is small enough to measure accurately the voltage changes but small enough not to affect the circuit. From this, the average current could be calculated by integrating the area under the voltage curve produced.

6. Discussion and conclusions

6.1. Discussion

To be confident that Bluetooth is a feasible WC protocol it needed to achieve suitable power consumption, distance, data speed and data volume performance levels. The work done to create a working prototype strongly supports the conclusion that BLE would work although two of the performance criteria need comment and further work be certain of this.

Firstly, the power consumption was higher than the values presented in the "Opportunistic Sensor Data Collection with Bluetooth Low Energy" article [10]. This was because the readings taken in that article were for average current consumption, while the method used here only allowed for peak current consumption to be used. At present the energy consumption in this prototype is too high to be implemented into a wearable device, however it should be possible to reduce current consumption to allow BLE to be used. This project identified the following steps that could be taken to try and reduce the current consumption sufficiently,

- Use a newer version of BLE choosing from a selection of more customisable BLE modules to reduce the current consumed
- Adapt the code which is run by the chips to optimise the power usage
- Customise the advertising structure

Secondly, the reliability of the communication was not 100% since this depended on the advertising interval. There is a balance to be found between increasing the advertising interval, to increase the probability of a sensor being detected, and the increase in power this requires. In wearable sensor application where updates are not required every second, a slower advertising interval may suffice although the option depends on the end use application. The ability to adjust the parameter of the BLE advertising mode to suite the devices application is one of its strengths.

6.2. Future Project development

This section will cover where this project would go if more time, money and resources were available.

With a short increase in time further research with the current prototype would be done to examine the limits of the current system, i.e. how many sensors could be handled by the master module and how energy efficient the module could be made with adaptions to the code. The advertising modules would be put into sleep mode during ADC readings and advertising mode turned on and off between transmissions. To reduce redundancy the sensors would not transmit if no temperature change was measured.

It is expected that the increased time it would take for the master to process all the slaves would increase power consumption and, therefore, reduce the operating time of the master module. This could be handled by changing the string that sensors are recognised by to the first 4 bytes of the UUID as these will take less time to read. The minor field could then be found using an offset from the end of the recognised UUID rather than extracting everything between two ‘colons’ and then, later, extracting the data from that. This would save time and, therefore, the energy used in the master module allowing it either to scan more slave modules or return to lower power mode more quickly. Only scanning for 4 bytes, however, reduces the number of slave modules that can be produced and increases the chance of the error as the byte’s arrangement is more likely to appear elsewhere. Experimentation and analysis of these trade-offs would be required. Energy efficiency could also be achieved by experimenting with the Sleep command and how variations of its application in the code could reduce power consumption. Data processing could also be moved back to the cloud to reduce processing requirements on the master module.

Moving past the current set up, with a greater budget the RN4871 and its breakout board would be purchased and experimented with to create a wearable sensor prototype for end use application. RN4871 would have the advantage of having a built in MCU and, therefore, have a smaller formfactor and a lower power consumption. The advertising packets would still be used but they would have greater versatility due to the improved Bluetooth 4.2 specification. Creating a custom advertisement structure using the 0xFF data type, which is manufacture specific, would also save energy as the packets being advertised could be made smaller.

Having contacted SIG about the Bluetooth communication in this project, their recommendation is to look at their resources about Bluetooth mesh networks. This could lead to a second method to connect the slave modules to a master module and would require investigation to compare its energy efficiency to the advertising mode investigated in this project.

6.3. Conclusion

The prototype developed in this project provides proof of concept that BLE’s advertising mode can be used as a low power communication protocol in a WPAN to connect multiple sensors to a master module. Recording peak current consumptions of 8.8mA in the most power efficient AT settings and with options to further increase sensor modules power efficiency. There is a trade-off to be made, depending on the application, between reliability in advertising frequency and current consumption. BLE has promising uses in wearable sensor applications. Future development of this project with BLE version 4.2 or 5 appear to promise a reduction in scale, an increase in reliability and reduction in power consumption.

7. Project management, consideration of sustainability and health and safety

7.1. Project Management

Several tools were used to keep this project on track and prevent time from being wasted. The Gantt chart, shown in figure 22, was implemented to keep the tasks in the project moving and under control. The Gantt chart was updated as the project went along, with tasks being updated when they over ran or finished early. Meetings were organised with the project supervisor to keep the work on track and to discuss the quality of the work while keeping it on point. Meetings with academic support were also organised to check the structure of the report written and the quality of the English used in it. The combination of tools used above helped this project meet its deadlines and ensure the quality of its work.

ID		Task Mode	Task Name	Duration	Start	Finish
1			**Research wireless communication technologies**	**108.83 hrs**	**Mon 24/09/18**	**Tue 27/11/18**
2			Research LoRa (LPWAN)	14.5 hrs	Fri 19/10/18	Fri 26/10/18
3			Research Sigfox (LPWAN)	14.5 hrs	Fri 19/10/18	Fri 26/10/18
4			Research NB-IoT (LPWAN)	14.08 hrs	Fri 19/10/18	Fri 26/10/18
5			Research BLE (WPAN)	14.5 hrs?	Tue 30/10/18	Tue 06/11/18
6			Research ZigBee (WPAN)	14.5 hrs?	Tue 30/10/18	Tue 06/11/18
7			Research Wi-Fi (LAN)	11.67 hrs?	Wed 07/11/18	Wed 14/11/18
8			Wireless communication analysis	5.08 days	Fri 16/11/18	Tue 27/11/18
9			Literature Review	36.83 hrs?	Wed 28/11/18	Sun 13/01/19
10			**Preliminary design**	**39.67 hrs?**	**Fri 07/12/18**	**Tue 22/01/19**
11			Software requirements	47.5 hrs?	Fri 07/12/18	Tue 12/03/19
12			Hardware requirements	39.67 hrs?	Fri 07/12/18	Tue 22/01/19
13			Component research	14.5 hrs?	Tue 15/01/19	Tue 22/01/19
14			**Preliminary Report due**	0 hrs	Fri 14/12/18	Fri 14/12/18
15			5 page report	29.58 hrs	Tue 27/11/18	Fri 14/12/18
16			component order	0 hrs	Tue 22/01/19	Tue 22/01/19
17			**Prototyping**	**96.58 hrs**	**Fri 25/01/19**	**Tue 02/04/19**
18			Implementation of design 1	19.33 hrs?	Fri 25/01/19	Wed 13/02/19
19			**Implementation of design 2**	**27.5 hrs**	**Fri 15/02/19**	**Tue 26/02/19**
20			Component research and ordering	4 hrs?	Fri 15/02/19	Sun 17/02/19
21			prototyping	14.5 hrs?	Tue 19/02/19	Tue 26/02/19
22			**Implementation of design 3**	**31.17 hrs**	**Tue 26/02/19**	**Fri 15/03/19**
23			PCB design	18.92 hrs?	Tue 26/02/19	Fri 08/03/19
24			PCB manufacture	5.83 hrs?	Fri 08/03/19	Sun 10/03/19
25			PCB and chip testing	8.25 hrs?	Mon 11/03/19	Fri 15/03/19
26			**Implementation of design 4**	**32.75 hrs**	**Fri 15/03/19**	**Sun 31/03/19**
27			Software design	9.67 hrs?	Fri 15/03/19	Tue 19/03/19
28			Software implementation	14.5 hrs?	Tue 19/03/19	Tue 26/03/19
29			Ordering components	4 hrs?	Tue 26/03/19	Wed 27/03/19
30			Proof of concept	6.83 hrs?	Wed 27/03/19	Sun 31/03/19
31			**Final report due**	0 hrs	Wed 08/05/19	Wed 08/05/19
32			Final report	40.25 hrs?	Mon 01/04/19	Fri 26/04/19
33			interview preparation	15.5 hrs?	Tue 21/05/19	Wed 29/05/19
34			**Interview period**	10.67 hrs	Fri 24/05/19	Wed 29/05/19

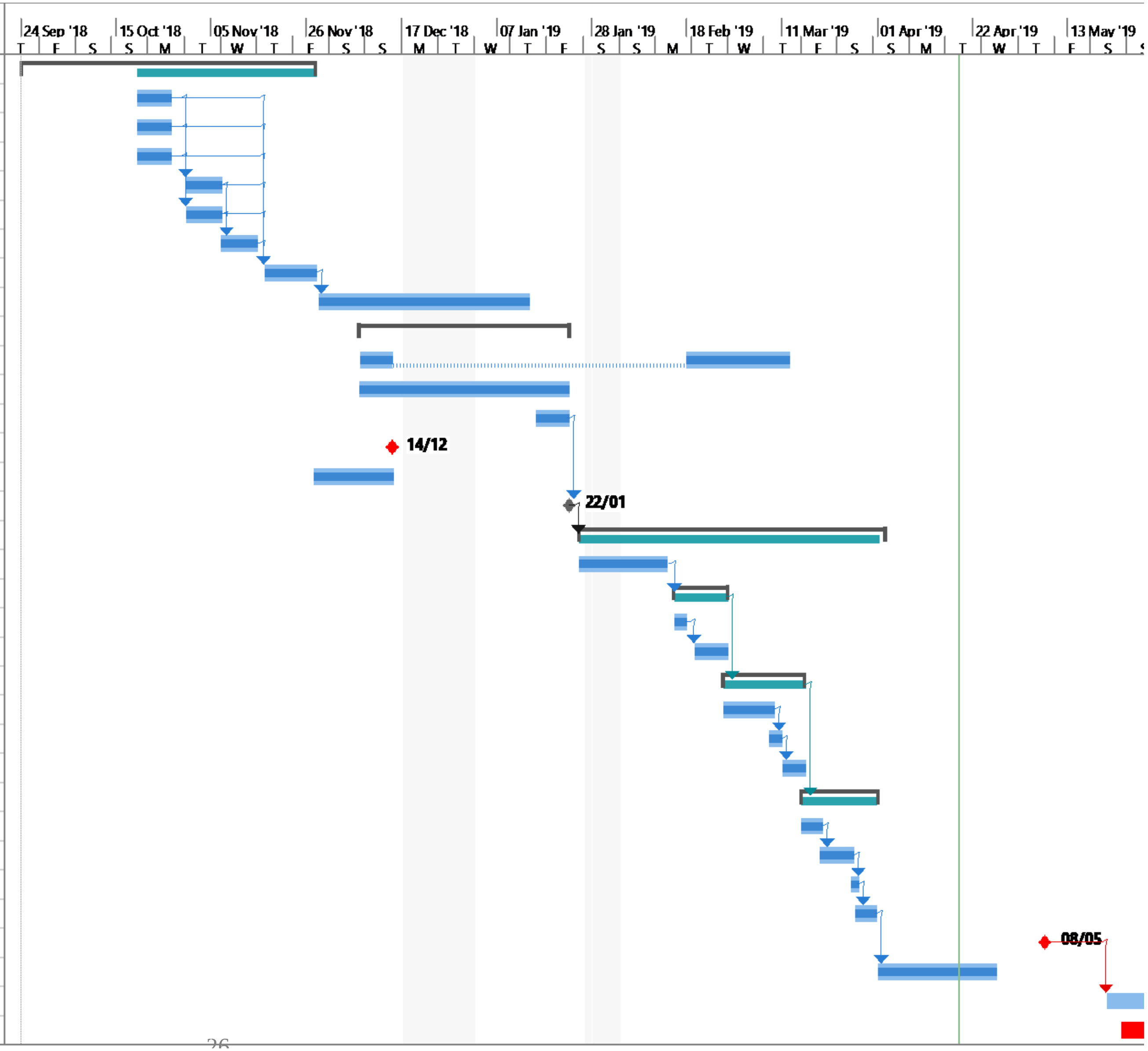


Figure 22: Gantt chart used to keep the project on track

7.2. Project Risk Assessment

A number of components of this project risked delaying the project and causing it to miss its deadlines. These components had to be identified early and assessed to reduce the probability of them disrupting the project. These components were assed in table 2.

Table 2: Risk Assessment of Project

Ref:	Risk Item	Cause	Effect	Likelihood (1low-5high)	Severity (1low-5high)	Risk Score (1low-25high)	Action to control risk	Contingency Plan
A	Delayed shipment of parts	Delay in supply	Delay in manufacture of protype	2	2	4	Parts ordered once parts confirmed from reputable sources to reduce chance of delay. Have other work planned over delivery window so that time isn't wasted.	Find other areas of the project to work on to prevent wasting time
B	Damaged parts	Static charge builds up, excess heat, wired with the wrong polarity.	Delay in manufacture. Go over budget.	4	4	16	Parts will be touched as little as possible before installation and kept in static protective environments. The Soldering Iron will be operated under magnifying lenses to improve accuracy and Peter Armitage will do the delicate soldering to rely on his expertise.	Alternative parts and method planned to be ordered to act as a backup to the delicate modules.
C	Lab unavailable	Lack of session planning	Unable to use Lab for soldering or its equipment	1	2	2	Lab times will be planned and checked with Peter Armitage ahead of time to ensure the Labs availability.	Relevant software will be downloaded onto home computers and equipment sourced encase its needed.

7.3. *Health and Safety*

A risk assessment of potential hazards was carried out to reduce the chance of personal injury or damage to University equipment. The Universities' risk assessment form was used as a template. The Risk assessment can be seen in table 3 below.

Table 3: Risk Assessment form

Ref:	Hazard	Who may be harmed?	Cause	Consequence	Likelihood (1 low – 5 high)	Severity (1 low – 5 high)	Risk Score (1 low– 25 high)	Action to control risk	Contingency
A	Electrocution	Student	Touching exposed pins on plug of soldering iron	Personal Injury/death	1	5	5	Hold plug by grip never touching pins. Don't turn on socket till plug properly fitted. A valid electrical test certificate will be required for soldering irons. Soldering will only be done in the lab were circuit protection exists.	Location of first aid kit always noted and secondary person required in the lab.
B	Cut from wire cutting equipment	Student	Misuse of equipment	Personal Injury	2	2	4	Extra care must be taken at all time when working with these tools.	Location of first aid kit noted and stocked with plasters.
C	Burn from soldering iron	Student	Misuse of equipment	Personal Injury	3	2	6	Extra care must be taken at all time when working with these tools. Clear working area to reduce chance of obstruction. Ensure proper soldering holder is within easy reach.	Note location of cold water tap, first aid kit.
D	Fire from soldering iron	Student	Misuse of equipment	Fire damage. Personal injury/death	1	5	5	Working area cleared of non-vital equipment. Note location of fire equipment before starting. Lab technician should be notified before starting. Soldering Iron will be turned off when not in use.	Note location of fire equipment and alarm and check access to emergency exit

7.4. *Environmental Considerations*

The environmental impact of this project was considered throughout. However, with a small budget only limited precautions could be taken. Due to the development of this project more components and, therefore, resources were used than ideally liked.

In future, further time would be spent researching and discussing proposed solutions with people who have relevant experience to prevent the unnecessary purchase of components. Where possible, components were sourced renewably, for example the resistors used were acquired from the lab and will be returned after use to allow them to be reused. Doing so has reduced the environmental impact of the project. Due to the small budget however, cheaper components had to be ordered from China increasing the project's carbon footprint. If components are sourced locally it allows not only their carbon footprint to be lower but also helps strengthen the local economy. Drawbacks of sourcing components locally include increased expenditure due to higher minimum wage in the west. This increases the final price of products and increase the cost of R&D. However, from a sustainability viewpoint it is advantageous.

Due to the miniscule design of electronic components and the range of material which can be built onto a custom PCB, it is very difficult to recycle components and elements. There are, however, some processes which can be used to recycle components at their end of life stage. Gasification and plasma treatment are used to extract metal from residual materials under high temperature releasing excess gases from waste materials (e.g. plastics) [26]. The overall effects of the process need to be considered as large amounts of energy is required to achieve these processes as well as handling of gas waste. The PCB that was manufactured for the RN4871 was made as small as functionally possible for the prototyping process to reduce initial material usage. PCB's are poor products to recycle as only the copper traces are recoverable and are required in large quantities to be economically viable [27]. The recovery process uses chemical leaching to remove the copper. If the product were to go into manufacture the whole product life cycle would be considered. Furthermore, the modules would need to be designed for longevity. To ensure this, it would be included in the design criteria. The graphene temperature sensors could be detachable from the communication module to allow the graphene to be recycled back into raw graphene for reuse if it gets damaged, thus allowing the communication module to be connected to a new sensor. Ideally the power for the modules would be from a renewable source. With further research it may be possible to find an alternative power source for the modules other than coin cell batteries which have a large environmental footprint attached to them due to the raw materials usage, their short life span and their limited end of life options.

References

[1] IDC (2018) Forecasts Slower Growth for Wearables in 2018 Before Ramping Up Again Through 2022 [online]. Framingham: IDC. Available from: https://www.idc.com/getdoc.jsp?containerId=prUS44276818 [Accessed29.09.18]

[2] Siekkinen, M., hiienkari, M., Nurminen, J. and Nieminen, J. (2012), How low energy is Bluetooth low energy? Comparative measurements with ZigBee/802.15.4, *2012 IEEE Wireless Communication and Networking Conference Workshop (WCNCCW)*, IEEE, Paris, 01 April 2012. IEEE Xplore Digital Library [online].Available from: https://ieeexplore.ieee.org/abstract/document/6215496 (Siekkinen. M et al, 2012) [Accessed 31.09.18]

[3] Centenaro, M., Vangelista, L., Zanella, A and Zorzi, M. (2016) Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios [online] . *IEEE Wireless Communications.* 23 (5),pp. 60-67. [Accessed 19.10.18]

[4] Mahmood, A., Javaid, N. and Razzaq, S.(2015) A review of wireless communications for smart grid. *Renewable and Sustainable Energy Reviews* [online]. 41, pp. 248-260. [Accessed 19.10.18]

[5] Bluetooth SIG (2019) *Our History*. Available from: https://www.bluetooth.com/about-us/our-history [Accessed 24.10.18]

[6] Bisdikian, C . (2001) An overview of the Bluetooth wireless technology [online]. *IEEE Communications Magazine.* 12, pp. 86-94. [Accessed 24.10.18]

[7] Ohmura, R., Naya, F., Noma, H. and Kogure, K. (2006) B-packed: a Bluetooth-based wearable sensing device for nursing activity recognition [online]. *2006 1st international Symposium on Wireless Pervasive Computing*, IEEE, Phuket, 16-18 Jan. 2006. IEEE Xplore Digital Library [online]. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1613628&tag=1 [Accessed 25.10.18]

[8] Collotta, M., Pau, G., Talty, T and Tonguz, O. (2018) Bluetooth 5: A Concrete Step Forward towards the IoT [online]. *IEEE Communications Magazine*. 56 (7), pp 125-131. [Accessed 26.10.18]

[9] Yao, J., Schmitz, R. and Warren, S. (2005) A wearable point-of-care system for home use that incorporates plug-an-play and wireless standards, *IEEE Transactions on Information Technology in Biomedicine* [online]. 9 (3), pp363-371. [Accessed 28.10.18]

[10] Aguilar, S,. Ferre, R, and Gomez, C. (2017) Opportunistic Sensor Data Collection with Bluetooth Low Energy. *Sensors* [online]. 17 (1). [Accessed 04.11.18]

[11] Gomez, C,. Oller, J. and Paradells, J. (2012) Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. *Sensors [online]. 12, pp. 11734-11753.* [Accessed 08.11.18]

[12] Argenox (N/A) *BLE ADVERTISING PRIMER*. Available from: https://www.argenox.com/library/bluetooth-low-energy/ble-advertising-primer/#ble-advertising-packets [Accessed08.11.18]

[13] Warne, W. (2017) Bluetooth Low Energy- it starts with Advertising. *Bluetooth Blog* [Blog]. 15 February. Available from:

https://blog.bluetooth.com/bluetooth-low-energy-it-starts-with-advertising [Accessed 14.01.19]

[14] Microchip (N/A) *RN4871 Bluetooth Low Energy Module*. Available from: https://www.microchip.com/wwwproducts/en/RN4871 [Accessed 15.01.10]

[15] Microchip Technology. (2018) *RN4870/71 Bluetooth Low Energy Module User's Guide* [online]. Report number: DS50002466C. Microchip. Available from: http://ww1.microchip.com/downloads/en/DeviceDoc/RN4870-71-Bluetooth-Low-Energy-Module-User-Guide-DS50002466C.pdf [Accessed 16.01.19]

[16] Argenox (2019) *A guide to selecting A Bluetooth Chipset.* Available from: https://www.argenox.com/bluetooth-low-energy-ble-v4-0-development/library/a-guide-to-selecting-a-bluetooth-chipset/ [Accessed 14.01.19]

[17] Develop EX (2017) *Most popular BLE controllers.* Available from https://developex.com/blog/most-popular-ble-controllers/ [Accessed]

[18] JNHuaMao Technology Company (2014) *Bleutoooth 4.0 BLE module Datasheet* [online]. Report Number: V524. JNHuaMao Technology Company. Available from: http://fab.cba.mit.edu/classes/863.15/doc/tutorials/programming/bluetooth/bluetooth40_en.pdf [Accessed 16.01.19]

[19] Laird Connectivity. (N/A) *Datasheet LoRa/Bluetooth Low Energy (BLE) Module* [online]. Report number: 1.9. Laird Connectivity. Available from: https://www.mouser.co.uk/datasheet/2/223/CS-DS-RM1xx_v1_9-1536676.pdf [Accessed 17.01.19]

[20] Arduino (N/A) *AEDUINO UNO REV3.* Available from: https://store.arduino.cc/arduino-uno-rev3[Accessed17.01.19]

[21] ANALOG DEVICES. (N/A) *Low Voltage Temperature Sensors TMP35/TMP36/TMP37* [online]. ANALOG DEVICES. Available from: http://ctms.engin.umich.edu/CTMS/Content/Activities/TMP35_36_37.pdf [Accessed 22.01.19]

[22] DFROBOT (N/A) *FireBeetle Board-328P with BLE 4.1* [online]. Available from: https://www.dfrobot.com/wiki/index.php/FireBeetle_Board-328P_with_BLE4.1_SKU:_DFR0492 [Accessed 02.02.19]

[23] Microchip Technology. (2018) *RN4870/71 Bluetooth Low Energy Module* [online]. Report number: DS50002489C. Microchip. Available from: http://ww1.microchip.com/downloads/en/DeviceDoc/50002489C.pdf [Accessed 16.01.19]

[24] Laird. (2018) What You Need to Know About Bluetooth SIG Deprecation of BT v4.1 and Older Specs. *Laird Wireless Connectivity Blog* [blog]. 13 July. Available from: http://www.summitdata.com/blog/need-know-bluetooth-sig-deprecation-bt-v4-1-older-specs/ [Accessed 15.02.19]

[25] Stoffregen, P. (2016) AltSoftSerial (1.4.0) [computer program]. Available from: https://www.arduinolibraries.info/libraries/alt-soft-serial [Accessed 19.03.19]

[26] Debnath, B., Roychowdhury, P. and Kundu, R. (2016) Electronic Components (EC) Reuse and Recycling- A New Approach towards WEEE management. *Prodedia Environmental Sciences*. 35, pp 656-668

[27] Newbury Electronics. (2019) CAN PRINTED CIRCUIT BOARDS BE REUSED OR RECYCLED?. *Newbury Electronics News.* 27 March. Available from:
http://www.news.newburyelectronics.co.uk/newbury-blog/can-printed-circuit-boards-be-reused-or-recycled/
[Accessed 05.04.19]

Appendix

1. APPENDIX A: TMP36 Sensor to I phone code

```
float VoltageValue, TempValue, TempValueC;
void setup() {
 pinMode(A0, INPUT);
 Serial.begin(9600);
}
void loop() {
 //voltage Value in mV
 VoltageValue = analogRead(A0);
 // Voltage supply / voltage levels (1024 (10 bit ADC))
 TempValue = VoltageValue * 0.004882814;
 /* Voltage off set from data sheet
  conversion from millivolts to volts and linear relationship
  of 10 mV = 1degree celcius */
 TempValueC = ((TempValue - 0.5) * 100.0);
 Serial.print("Temperature: ");
 Serial.print(TempValueC);
 Serial.println(" C");
 delay(10000);
```

2. APPENDIX B: Master module

```
#include <AltSoftSerial.h> //AltSoftSerial BTSerial;

AltSoftSerial BTSerial(8, 9); // Pins: RX=8, TX=99
// AT commands and responses for HM10/11
char* ATcmds[] = {"AT", "AT+RENEW", "AT+IMME1", "AT+ROLE1"};
char* ATresps[] = {"OK", "OK+RENEW", "OK+Set:1", "OK+Set:1"};
String response = ""; // ascii string from HM10/11
int SearchScan(String UUID);
int SendCmd(void);
void ReadScan(unsigned long duration);

void setup() {
 Serial.begin(9600); // Open serial communications
 BTSerial.begin(9600); //Open Bluetooth SoftSerial communication
 ReadScan(3500); // This will effectively flush the sSerial buffer
 Serial.println("Master Module (V3.2)");
}
void loop() {
 // fixed UUID string for temperature sensors
 String UUIDs [] = {"74278BDAB64445208F0C720EAF059935", "12345678B64445208F0C720EAF059935"};
 char* SenNam [] = {"Sensor1", "Sensor2"};
 int SensNum;
 if (SendCmd()) {
   Serial.println("Error sending AT commands");
   ReadScan(3500); // This will effectively flush the sSerial buffer
   delay(1000);
 }
 else {
   while (1) {
     BTSerial.write("AT+DISI?");   //scans for advertisers
     ReadScan(4000); // 5 seconds works well
     // uncomment the next line to print the entire response screen
     //Serial.println(response);
     if (BTSerial.overflow()) {
      Serial.println("** SoftwareSerial overflow! **");
     }
     // SearchScan() will do all the work and return either
     // 0 - no sensor in the scan.
     // 1 - sensor found.
     // Look for sensor with UUIB Declared.
     for (SensNum = 0; SensNum < 2; SensNum++) {
      switch (SearchScan(UUIDs [SensNum], SenNam [SensNum] )) {
        case 0:
         Serial.print(SenNam [SensNum]);
         Serial.println(" not found");
         //Serial.println("");
         break;
        case 1:
         Serial.print(SenNam [SensNum]);
         Serial.print(" found");
         Serial.println(":");
         break;
        default:
         Serial.println();
         Serial.println("!!!unknown Sensor!!!");
         break;
      }
     }
   }
 }
}
//------------------------------------------
// Send the AT commands to set up the HM10/11
// must return the correct responses
// ok if 0 is returned, error otherwise
int SendCmd(void) {
 int x, ecount;
```

```
  ecount = 0;
  for (x = 0; x < 4; x++) {
    BTSerial.write(ATcmds[x]); //sends command to Bluetooth module
    Serial.print(ATcmds[x]);
    ReadScan(1000);
    Serial.print("->");
    Serial.println(response);
    //checks repsonse against expected response
    if (response != ATresps[x]) {
      ecount++;
    }
  }
  BTSerial.write("ATDELO1"); // send ATDELO1 which does not have a response
  delay(500);
  return (ecount);
}
//-------------------------------------------
// Read the response string from the HM10/11 stays for duration (seconds) to get a complete string
void ReadScan(unsigned long duration) {
  char chr;
  response = "";
  // for timer
  unsigned long starttick = 0;
  unsigned long curtick = 0;

  //Stays here for duration of time reading the software serial
  starttick = millis();
  while ( ( (curtick = millis() ) - starttick) < duration )
{
    while (BTSerial.available() != 0) {
      chr = BTSerial.read();
      response.concat(chr);
    }
  }
}
//-------------------------------------------
int SearchScan(String UUID, char* Name) {
  int UUIDIndex = 0;
  int supdate = 0;
  char SsenorTEM[] = {"0000"};
  char SsensorID[] = {"0000"};
  float TempC, Sensor1Temp, Temp;
  String sTemp, sID;
  unsigned long sensorTemp, sensorID;
  int sliceStart, sliceEnd;
  String sliceP2;
  if ((UUIDIndex = response.indexOf(UUID)) == -1)
{
    return (supdate);
  }
  // a sensor UUID was found and uIndex points to
  // the begining of the string - search for the next ':'
  //marks start of Data field
  if ((sliceStart = response.indexOf(':', UUIDIndex))
== -1) {
    return (supdate);      // if no colon found assume error and dump the string
  }
  // get the next colon
  //marks start of temperature data
  if ((sliceEnd = response.indexOf(':', sliceStart + 1))
== -1) {
    // if no colon found assume error and dump the string
    return (supdate);
  }
  // get the substring bounded by the two colons
  sliceP2 = response.substring(sliceStart + 1,
sliceEnd);
  if (strlen(sliceP2 != 10)) {
    // P2 needs to be exactly 10 characters, if not, assume error and dump the string
    return (supdate);
  }
    for (int i = 0; i < 7; i++) //seperates the relevant characters into a substrings
  {
    Serial.print(Name[i]);
  }
  Serial.println("");
  sID = sliceP2.substring(0, 4);
  sTemp = sliceP2.substring(6, 8);
  sID.toCharArray(SsensorID, 5);   // copy them to their constant char analogs
  sTemp.toCharArray(SsenorTEM, 4);
  Serial.print("Hex value:");
  Serial.println(sTemp);
  sensorTemp = strtoul(SsenorTEM, NULL, 16);   // get hex ascii string to unsigned longs
  Temp = sensorTemp * 0.004882814;
  TempC = ((Temp - 0.5) * 100.0);
  Serial.print("Temperature:  ");
  Serial.print(TempC);
  Serial.println(" Degrees Celsius");
  return (1);
}
```

3. APPENDIX C: Sensor module

```
// Most update working code 23/04/2019
#include <AltSoftSerial.h> //AltSoftSerial BTSerial;

AltSoftSerial BTSerial(8, 9); // Pins: RX=8, TX=9
int pos = 0;
char c = ' ';
char reply[30];
char Mino[] = {"0000"};
char MinoResps;
String HexTemp, MinoTemp, MinoRespStr;
String MinoPre = "AT+MINO0x00";
String MinoRespsPre = "OK+Set:00";
String response = ""; // ascii string from HM10/11
// AT commands and responses for HM10/11
char* ATcmds[] = {"AT", "AT+RENEW", "AT+RESET", "AT", "AT+POWE0",
```

```
"AT+MARJ0x1234", "AT+MINO0xFA01",
"AT+IBE074278BDA", "AT+ADVIF",
"AT+NAMEsensor1", "AT+ADTY3",
"AT+IBEA1", "AT+DELO2", "AT+RESET"};
//7000ms advertising interval
char* ATresps[] = {"OK", "OK+RENEW",
"OK+RESET", "OK", "OK+Set:0",
"OK+Set:0x1234", "OK+Set:0xFA01",
"OK+Set:0x74278BDA", "OK+Set:F",
"OK+Set:sensor1", "OK+Set:3", "OK+Set:1",
"OK+DELO2", "OK+RESET"};
void ReadScan(unsigned long duration);
String decToHex(unsigned int decValue, byte
desiredStringLength);
int SendCmd();
int SendData();
//------------------------------------------------------------
--------------
void setup() {
  Serial.begin(9600);
  Serial.print("Sketch: ");
Serial.println(__FILE__);
  Serial.print("Uploaded: ");
Serial.println(__DATE__);
  Serial.println(" ");
  pinMode (A0, INPUT);
  BTSerial.begin(9600);
  Serial.println("BTserial started at 9600");
  Serial.println(" ");
  ReadScan(3500); // This will effectively flush the
BTSerial buffer
}
//------------------------------------------------------------
--------------
void loop() {
  int VoltageValue;
  if (SendCmd()) {
    Serial.println("Error sending AT commands");
    ReadScan(3500); // This will effectively flush the
sSerial buffer
    delay(1000);
  }
  else {
    while (1) {
      VoltageValue = analogRead(A0); //Voltage
reading from thermistor
      Serial.print("ADC reading:");
      Serial.println(VoltageValue);
      HexTemp = decToHex(VoltageValue, 2);
//String(VoltageValue, HEX);
      HexTemp = CaseChk(HexTemp);
      delay(500);
      MinoTemp = MinoPre + HexTemp;
      MinoRespStr = MinoRespsPre + HexTemp;
      MinoRespStr.toCharArray(MinoResps, 14);
      MinoTemp.toCharArray(Mino, 14);
      SendData();
      delay(5000);
      // Currently won't take mV readings over 255 as
that will require 3 Hex digits.
    }
  }
}
//------------------------------------------------------------
--------------
// Read the response string from the HM10/11. stays
for duration (seconds) to get a complete string
void ReadScan(unsigned long duration) {
  char chr;
  response = "";
  // for timer
  unsigned long starttick = 0;
  unsigned long curtick = 0;
  //Stays here for duration of time reading the
software serial
  starttick = millis();
  while ( ( (curtick = millis() ) - starttick) < duration )
{
    while (BTSerial.available() != 0) {
      chr = BTSerial.read();
      response.concat(chr);
    }
  }
}
//------------------------------------------------------------
--------------
int SendCmd() {
  int x, ecount;
  ecount = 0;
  for (x = 0; x < 13; x++) {
    BTSerial.write(ATcmds[x]); //sends command to
Bluetooth module
    Serial.print(ATcmds[x]);
    ReadScan(1500);
    Serial.print("->");
    Serial.println(response);
    //checks repsonse against expected response
    if (response != ATresps[x]) {
      Serial.println(ecount); //Debugging line
      ecount++;
      Serial.println(ecount); //Debugging line
    }
  }
  // send ATDELO1 which does not have a response
  BTSerial.write("ATDELO1");
  delay(500);
  return (ecount);
}
//-----------------------------------------------------------
String decToHex(unsigned int decValue, byte
desiredStringLength) {
  String hexString = String(decValue, HEX);
  while (hexString.length() < desiredStringLength)
hexString = "0" + hexString;
  return hexString;
}
//------------------------------------------------------------
---------
int SendData() {
  int x, ecount;
  ecount = 0;
  BTSerial.write(Mino);
  Serial.print(Mino);
  ReadScan(1000);
  Serial.print("->");
  Serial.println(response);
  if (response != MinoResps) {
    //Debugging line
    Serial.println(ecount);
    ecount++;
    //Debugging line
```

```
    Serial.println(ecount);
  }
  delay(500);
  return (ecount);
}
//-----------------------------------------------------
String CaseChk( String Hex) {
  for (int x = 0; x < 2; x++) {
    if (Hex[x] == 'a') {
      Hex [x] = 'A';
    }
    else if (Hex[x] == 'b') {
      Hex [x] = 'B';
    }
    else if (Hex[x] == 'c') {
      Hex [x] = 'C';
    }
    else if (Hex[x] == 'd') {
      Hex [x] = 'D';
    }
    else if (Hex[x] == 'e') {
      Hex [x] = 'E';
    }
    else if (Hex[x] == 'f') {
      Hex [x] = 'F';
    }
  }
  return (Hex);
}
```