JAMES COOK UNIVERSITY

SCHOOL OF ENGINEERING & PHYSICAL SCIENCES

EG4011

Voice activated system for the disabled

Kyyle Newton

Thesis submitted to the School of Engineering in partial fulfilment of the requirements for the degree of

Bachelor of Engineering

(Electrical & Electronics)

October 2012

Statement of Access

I, the undersigned, the author of this thesis, understand that James Cook University will make it available for use within the University Library and, by microfilm or other means, allow access to users in other approved libraries. All users consulting with this thesis will have to sign the following statement:

In consulting this thesis I agree not to copy or closely paraphrase it in whole or in part without written consent of the author; and to make proper public written acknowledgement for any assistance which I have obtained from it.

Beyond this, I do not wish to place any restriction on access to this thesis.

Kyyle Newton Date

Sources Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Kyyle Newton Date

Acknowledgements

I would like to acknowledge Dr Owen Kenny as my supervisor throughout the year for the course of this project. Given special thanks to him for the time and effort spent given me assistance and guidance towards the design and completion of this project. As well as for going through the effort of reading through my drafts and providing quality feedback, no matter how late and untimely they were submitted. I also wish to take a moment to thank my housemates and friends who put up with me while I completed this project and who I forced to read sections of this document.

Abstract

For many people with disabilities even some of the most common daily actions, which unaffected people take for granted, can be very difficult. Not being able to easily perform simple tasks such as opening doors, turning on the lights or making phone calls can severely impair their day to day life. This was due to the complications caused by each of their disadvantages and disabilities. The goal of this project was to design a voice automated system that was able to provide aid to a large range of disabled people.

The goal as to form a design that was reasonably cost effective, modular, adaptive, easy to implement, easy to operate, has a non-invasive installation, and was suitable for operation by people suffering from the widest range of problems.

This was achieved by designing a PC program that utilised Dynamic Time Warping (DTW) for speech recognition and a small non-invasive wireless hardware node capable of turning devices on and off. The node was designed to be able to communicate with the PC via Bluetooth communication and received commands for operation depending on recognised voice commands. The PC program was designed to receive, read and record voice commands wirelessly via a Bluetooth headset and then use this data for command recognition purposes. The PC program was also capable of switching devices controlled by the node on or off using a menu system.

Table of Contents

[Chapter 1: Table of Figures 7](#_Toc337197111)

[Chapter 2: Statement of Intent 9](#_Toc337197112)

[2.1 Introduction 9](#_Toc337197113)

[2.2 Project Aims 9](#_Toc337197114)

[2.3 Project Significance 10](#_Toc337197115)

[Chapter 3: Literature Review 11](#_Toc337197116)

[3.1 Existing Home Automation Solutions (HAS) 11](#_Toc337197117)

[3.2 System Control Methods 14](#_Toc337197118)

[3.3 Automatic Speech Recognition (ASR) 15](#_Toc337197119)

[3.4 Conclusion 16](#_Toc337197120)

[Chapter 4: Initial Methodology 17](#_Toc337197121)

[4.1 Overview 17](#_Toc337197122)

[4.2 Procedure 18](#_Toc337197123)

[4.2.1. Wireless Switching Adapter 18](#_Toc337197124)

[4.2.2. Network Hub & Automatic Speech Recognition (ASR) 19](#_Toc337197125)

[4.2.3. Smart Phone Integration 19](#_Toc337197126)

[4.3 Experiments 20](#_Toc337197127)

[4.3.1. Experimental Scope 20](#_Toc337197128)

[4.3.2. ASR Voice Recognition Experiment & programming 20](#_Toc337197129)

[4.3.3. Hardware Switching Experiment 20](#_Toc337197130)

[4.3.4. Sensor Input & Bluetooth Communications 21](#_Toc337197131)

[4.3.5. Smart Phone Control 22](#_Toc337197132)

[4.4 Progress (at time of proposal completion) 22](#_Toc337197133)

[4.5 Risk Assessment 23](#_Toc337197134)

[Chapter 5: Design Results 24](#_Toc337197135)

[5.1 Pc Side of Project 24](#_Toc337197136)

[5.1.1. Audio File Handling 24](#_Toc337197137)

[5.1.2. Pc Comms Port Monitoring and Communications 26](#_Toc337197138)

[5.1.3. Dynamic Time Warping 28](#_Toc337197139)

[5.1.4. Audio Capture 30](#_Toc337197140)

[5.1.5. User Interface and Debugging 31](#_Toc337197141)

[5.1.6. Complete PC Side Operation 33](#_Toc337197142)

[5.2 Node Side of Project 34](#_Toc337197143)

[5.2.1. Components 34](#_Toc337197144)

[A. Node Diagrams 34](#_Toc337197145)

[B. Microcontroller (MPS430G2553) 35](#_Toc337197146)

[C. Bluetooth Module 42](#_Toc337197147)

[D. Relay 45](#_Toc337197148)

[5.2.2. Communications 48](#_Toc337197149)

[5.3 Operation 50](#_Toc337197150)

[Chapter 6: Experimental Results 51](#_Toc337197151)

[6.1 Audio Capture 51](#_Toc337197152)

[6.2 DTW Tests 52](#_Toc337197153)

[6.2.1. Exact Match 53](#_Toc337197154)

[6.2.2. No Match 54](#_Toc337197155)

[6.2.3. Delay 55](#_Toc337197156)

[6.2.4. Amplitude Change 57](#_Toc337197157)

[6.2.5. Tempo Decrease 59](#_Toc337197158)

[6.2.6. Tempo Increase 61](#_Toc337197159)

[6.2.7. Speed decrease 63](#_Toc337197160)

[6.2.8. Speed Decrease 65](#_Toc337197161)

[6.2.9. DTW Results 66](#_Toc337197162)

[6.3 Switching 67](#_Toc337197163)

[Chapter 7: Discussion and Conclusions 68](#_Toc337197164)

[7.1 Functionality 68](#_Toc337197165)

[7.2 Usability 69](#_Toc337197166)

[7.3 Market 70](#_Toc337197167)

[7.3.1. Recommendations 70](#_Toc337197168)

[7.4 Conclusion 72](#_Toc337197169)

[Chapter 8: References 73](#_Toc337197170)

Document Word Count: 17,916

# Table of Figures

[Figure 1– Wav file selecting Graphical User Interface 24](#_Toc337197171)

[Figure 2 – Austain Powers voice data plotted from the sampled data 25](#_Toc337197172)

[Figure 3 – PuTTY terminal configuration screen, connecting to COM9 27](#_Toc337197173)

[Figure 4 – First version of comm interface 28](#_Toc337197174)

[Figure 5 – Lowest cost path of two identical waveforms 30](#_Toc337197175)

[Figure 6 – Bluetooth headset used for audio command capture 31](#_Toc337197176)

[Figure 7 – User interface 32](#_Toc337197177)

[Figure 8 – Wireless switching node bread board prototype 34](#_Toc337197178)

[Figure 9 – Power board implementation 34](#_Toc337197179)

[Figure 10 – T.I Launchpad & MSP430g2553 35](#_Toc337197180)

[Figure 11 – MSP430g2553 Block Diagram 35](#_Toc337197181)

[Figure 12 – CCS programing interface 37](#_Toc337197182)

[Figure 13 – Microcontroller main loop after initialisation 40](#_Toc337197183)

[Figure 14 - JY-MCU Bluetooth Module 42](#_Toc337197184)

[Figure 15 - Songle SRD-05VDC-SL-C relay 45](#_Toc337197185)

[Figure 16 – Blue Manager Bluetooth management suit node connection 48](#_Toc337197186)

[Figure 17 – Excerpt of PC logic checks for communications 49](#_Toc337197187)

[Figure 18 – PC command input 50](#_Toc337197188)

[Figure 19 – Audacity being used to edit the ‘Austain Powers voice sample’ 52](#_Toc337197189)

[Figure 20 – Two exact matched compared to each other 53](#_Toc337197190)

[Figure 21 – Two different audio samples compared to each other 54](#_Toc337197191)

[Figure 22 – Delay of -0.5s from original 55](#_Toc337197192)

[Figure 23 – Delay of -1s from original 56](#_Toc337197193)

[Figure 24 – Decrease of 10dB attenuation 57](#_Toc337197194)

[Figure 25 – Decrease of 20dB attenuation 58](#_Toc337197195)

[Figure 26 – Decreased tempo by 20 percent 59](#_Toc337197196)

[Figure 27 – Decreased tempo by 40 percent 60](#_Toc337197197)

[Figure 28 – Increased tempo by 20 percent 61](#_Toc337197198)

[Figure 29 – Increased tempo by 40 percent 62](#_Toc337197199)

[Figure 30 – Decreased speed by 20 percent 63](#_Toc337197200)

[Figure 31 – Decreased speed by 40 percent 64](#_Toc337197201)

[Figure 32 – Increased speed by 20 percent 65](#_Toc337197202)

[Figure 33 – Increased speed by 40 percent 66](#_Toc337197203)

[Figure 34 – Node breadboard design, results of a match with voice command 1 and menu options SwitchCommand3 and 4 beings sent to the controller. 67](#_Toc337197204)

[Figure 35 – Final version of the projects GUI 69](#_Toc337197205)

# Statement of Intent

## Introduction

For many people with disabilities even some of the most common daily actions, which unaffected people take for granted, can be very difficult. Not being able to easily perform simple tasks such as opening doors, turning on the lights or making phone calls can severely impair their day to day life. This was due to the complications caused by each of their disadvantages and disabilities. The goal of this project was to design and critically review a voice activated automation system for a disabled person’s home. The sounds used to trigger commands were to be completely user defined to accommodate persons with limited word sets and speech impediments. The focus of this work was be on the hardware of the automation system, and will continue to improve on the work by David Finch’s Remote Speech Activated Systems [[1](#_ENREF_1)]. The final design for this system was of a node which communicated with a PC wirelessly over Bluetooth capable of turning on or off automation devices that operate using power. The operation of the device was designed to be controller by speech input and recognition of specific words.

## Project Aims

There are several main aspects that were investigated for the design of this system, these included but were not limited to the supported automation processes, cost and availability of the hardware used and I/O communication characteristics. The aim of this project was to design an automation system that was easy to implement in a person’s home, competitively priced and versatile with potential uses.

The final goal of this project was to have a computer connected to a smart phone, at least one wireless power switching device and a wireless microphone, each connected via Bluetooth. The system was to be designed with the ability for additional wireless switching devices to be easily added to the system at later dates. Each wireless switching device was to be designed to connect as an intermediary between standard Australian three flat pin 10 amp power sockets (hence forth referred to as gpo’s) and the device running off of that gpo’s power. Each switching device was also to be designed to be capable of being connected to and transmitting data from a sensor.

Bluetooth connectivity was utilised for the communication between all devices in the system. The system was to be controlled via voice commands that were passed to the computer via the Bluetooth microphone. The computer was then used to evaluate the input and perform an action based of the outcome of that evaluation.

The interfacing method for the smartphone device was to be a command passing app, to allow remote voice control of some applications of phone such as making phone calls like 000 (Australian emergency number). A master control application would need to be designed to run on the hub computer to manage the states of the devices in the system. This part of the project was not undertaken due to time constraints.

## Project Significance

Research showed that currently for the majority [[2](#_ENREF_2)] of disabled or impaired individuals throughout Australia, any form of automation system was not a viable option. This was due to the significant costs [[3](#_ENREF_3)] involved in designing what were traditionally case by case designs, material costs, and the work needed for installation of these designs. These designs were most often hard wired into a building or residence and the automation process was controlled by a control panel. This project is significant in that the design allowed control of any appliance in the house powered by a gpo with the simple addition of a small adapter. This allowed for changes in what can be automated without the need of rewiring or designing anything at all. The same type of wireless switch could also be installed behind light switches to allow control of the lighting in a residence but would require the attention of a trained electrician. The significance of this systems design is that it could be implemented in any standard home, and additional units could be added as required or additional funds are available.

# Literature Review

There are many challenges faced by disabled individuals during their daily routine that the average person does not stop to consider. This is true both for those whose affliction is physical and those with mental or internal afflictions. Generally these kinds of people may need some combination of specially designed items, access or assistance to control and go about their day to day lives [[4](#_ENREF_4)]. These types of methods can include anything from specialised handles or counter weights on doors, ramp access or lift access to buildings [[5](#_ENREF_5)] [[6](#_ENREF_6)], specially designed bedrooms and bathrooms [[7](#_ENREF_7)] or specialist care takers [[4](#_ENREF_4)]. Studies in the UK [[8](#_ENREF_8)] [[9](#_ENREF_9)] [[10](#_ENREF_10)] have found that often due to the restrictions caused by a disability the afflicted find it harder to find work and are more likely to earn less money than the average citizen. This in turn causes an increase in poor living conditions in the homes of the disabled [[10](#_ENREF_10)] and an increased dependency on care and support from others. It is believed [[10](#_ENREF_10)] [[11](#_ENREF_11)] that these reasons contribute to the majority of disabled individuals in the UK living in rented properties rather than owning a home of their own. The increased dependency on others has been found to impact and in fact decrease the functionality of the dependant disabled person. Another UK study [[12](#_ENREF_12)] has shown that people who are completely dependent on others have a higher risk of suffering from the effects of depression and the risks associated with it. From this information it can be surmised that introducing a cost effective, non-invasive home automation system (HA or HAS) may not only increase the disabled person’s home living conditions but also increase their functionality, independence and mental health.

## Existing Home Automation Solutions (HAS)

There are several main forms of HA, with countless variations and changes in design to match case specific criteria. These forms of automation can be normally broken into two kinds of automation, wired and wireless. The Collins English Dictionary [[13](#_ENREF_13)] simply defines HA as “the control of domestic appliances by electronically controlled systems”. There are many noteworthy existing designs of HA, but these systems have mainly been designed for academia, large companies research divisions and the home garage hobbyist [[14](#_ENREF_14)]. There has yet to be the wide spread adoption of any particular HA design.

In the journal IEE Transactions on Consumer Electronics, A. R. Al-Ali and M. Al-Rousan discuss a method of automation [[15](#_ENREF_15)] that is very similar in principle to allot of industry automation systems but is implemented with the java language. This design uses an embedded control board such as a PLC that is physically connected to all sensors and control equipment in the system. One design [[15](#_ENREF_15)], uses a computer server to handle the integration of the systems components and at the same time offer the ability for remote access via the internet. The connection is made via Java protocols which have many built in data security features.

Some other distinctly different fixed line control systems utilise telephone, twisted pair Ethernet or home power lines to interface with the parts of the system. One design that utilises telephone line connections was designed by [[16](#_ENREF_16)] . This system would use dial tones sent over a phone connection rather than using the internet. The dial tone received would then trigger a pre-determined event, not all that different in principle to many over the phone services offered today.

Home power lines can be used as a connecting control structure for HA [[17](#_ENREF_17)] [[18](#_ENREF_18)] and is an innovative application for pre-existing infrastructure in homes. This method was designed primarily for turning on and off electronic devices throughout the home and is ideal for power management applications [[19](#_ENREF_19)]. The effectiveness of this system in a home is dependant largely on the number of separate power circuits wired throughout the home. Lighting and appliances with high power demand, such as air conditioners and hot water systems often have their own power circuit. It is now also possible to connect an internet router to you houses power lines and use them to transfer data [[19](#_ENREF_19)] so further development of applications, such as sensor feedback may be possible.

The second framework option for HA is wireless communication, which can be sub-categorised into four main different types of wireless communication. The four main types of communication that have been or are being implemented in the designs for wireless HA include designs based on, 3G mobile, Bluetooth, Wifi and ZigBee wireless protocols. Each of these communication methods can be used to as the framework for a HA system as opposed to the more traditional method of using wired communication methods. This gives the advantage of easier implementation in new homes, eliminating one of the biggest problems with the utilisation of wired methods. Furthermore, each of these four methods has additional advantages and disadvantages over each other, making some more suited to some tasks and design specifications then others.

R. Piyare and M. Tazil [[20](#_ENREF_20)] designed a HAS based on the Bluetooth protocol. Due to the nature of Bluetooth, the easiest method for setting up this network was to have a central ‘hub’ responsible for the analysis and control of the overall system. This hub in the case of [[20](#_ENREF_20)] was a cell phone that was connected via Bluetooth to other devices in the network. These devices could then be connected in turn to sensors or appliances. Bluetooth connections are very simple to set up and operate but one disadvantage of this method is the limited range of standard Bluetooth devices to approximately 10 metres. In the paper [[21](#_ENREF_21)] published in 2009 another disadvantage of utilising Bluetooth communications is that while a single sub-controller can be connected to multiple appliances, this can cause an access delay in communications. Since then performance on controllers and data handling has progressed to the point where streaming high bit rate music over Bluetooth is common place. While it is true that there would still be a delay with current technology, this delay is now insignificant unless constant large amounts of data need to be streamed at a reasonably high speed from several devices connected to a single sub-controller.

Wi-Fi and 3G wireless HA solutions are almost identical in design. The main exception is that the range of a Wi-Fi network can be as much as twice that of the Bluetooth network. The Wi-Fi design also allows for easy integration of remote and web based controls for the network. The 3G based solution, has a range as wide as the area of coverage for adequate 3G data transfers offered by the telecommunications provider. Both of these methods are capable of transmitting large amounts of data over their networks. The down side to these two devices is that the cost and complexity of required components is greater than that of the Bluetooth solution. Furthermore 3G communications require a sim card and plan with a telecom provider for each device.

The final main method being utilised in designs for wireless HA is the use of the ZigBee platform as discussed in [[21](#_ENREF_21)]. This solution proposed a combination of ZigBee communication and Wi-Fi based communication. The ZigBee platform is designed to use low intensity signals and small amounts of power [[21](#_ENREF_21)]. This is advantageous because it means that interference of wireless signals becomes less likely and the total cost of running the system (power consumption) is decreased. A significant disadvantage of the ZigBee platform is that the data transfer rate is low, up to a theoretical 250kbps maximum [[21](#_ENREF_21)]. This is enough for most control systems communications but not enough for some applications of sensor related management or multimedia networking. This is why an additional Wi-Fi network was proposed in the solution. This solution is designed with a ‘smart home’ in mind. Where all devices can give feedback of their states, have automated processes and multimedia can be streamed around the house. This system has a very wide range of capabilities but due to its complexity is also difficult to implement and is a case by case design.

## System Control Methods

There have been many methods of operator control implemented in control systems over the years with varying levels of success. Methods range from magnetic and resistive based glove solutions [[22](#_ENREF_22)], web based control pages, standard mouse operated GUI computer software, touch panel devices such as phones, voice controlled devices [[23](#_ENREF_23)] and even motion controlled devices such as the Microsoft Kinect [[24](#_ENREF_24)]. Due to the wide range of physical limitations that can be faced by disabled people, control solutions that require fine or specific motor control or a reasonable user understanding of potentially complex software for operation will be avoided for this design. This leaves two main forms of control, simple user friendly touch panel controls, or voice activated control methods.

The key benefit of voice controlled automation is that is can be wirelessly sent to the hub of the network and only requires the user to able to perform set sounds consistently. This implementation does not require much physical capability or control. All inputs in this method are completely hands free, making it a viable solution for people suffering from Tetraplegia [[25](#_ENREF_25)]. The disadvantage of this implementation is that it requires the user to remember and consistently perform specific sounds or phrases. This may be difficult or not possible at all for users with a form of disability that inhibits memory.

Touch panel controls are becoming more common place through society, as one of the main forms of interaction and input for devices such as phones, tablets and interactive information displays. This form of input is advantageous because a display can be placed behind it, making it intuitive to use. The display can be used to guide or inform the user of the controls, or define visual areas that can be ‘touched’ for different effects. The controls can be adaptive, unlike typical physical buttons, allowing for a case by case input method. Phones and tablet devices using this technology are common place now, and normally have both Wi-Fi and Bluetooth capabilities, they would be ideal to use as a wireless input method for an automation system. There are three key disadvantages of this system design. Firstly, the cost of such devices is high. The second disadvantage is that although intuitive and able to display prompts, this input method requires relatively fine motor control. These types of general movements or specific gestures might not be possible to be performed by some people with disabilities. The final disadvantage is the fragility of such electronics; drops and falls can easily damage or break them. Repairing or replacing these devices can have costs of several hundred dollars.

## Automatic Speech Recognition (ASR)

There are several methods of handling voice data used in applications of voice control. The methods of note that will be analysed for use in this system include the Hidden Markov Model (HMM) and Dynamic Time Warping (DTW). HMM is a statistical based model that is assumed to have unobservable states. This basically means that the process to get to a particular state is unknown, but the final output can be observed. By varying an input or feeding the output as an error signal back into the unknown states, a unique model that represents the actions of the unknown processes can be developed. The output of each unique input to an HMM can be represented by a unique vector. The most common method to utilise this for basic speech recognition is to sample speech in short intervals and send each sample through the HMM. The sum of the outputs produced will be a unique vector build from the combination of the inputs. In this way each combination of sounds can be individually identified. The DTW method of speech recognition is much more simple to implement, and is an approach that uses an algorithm to measure the similarities between two sequences. The nature of this type of recognition measures the similarities of transition, regardless of speed of the pattern or rate of change of a single sample. This is particularly useful in speech recognition as people do not always talk at the same speed.

## Conclusion

The goal is to form a design that is reasonably cost effective, modular, adaptive, easy to implement, easy to operate, has a non-invasive installation, and suitable for operation by people suffering from the widest range of problems. To do this the advantages and disadvantages of each topic covered in the literature review were reviewed with the aim of giving the best range of desired characteristics for the system. For the design to be modular, easily implemented and a non-invasive installation in someone’s home, a wireless automation system is best suited for the design. Hard wired automation systems can be expensive to install, remove or adjust and the people this design is aimed at often live in rented homes and have lower incomes. ‘Plug and Play’ style modules that simply need to be attached between a power source and the object to be controlled will allow for easy installation and huge variety of applications. Of the various wireless methods and design, Bluetooth connections are the simplest to handle, set up for an in experienced user and are cheap to implement. The ZigBee platform is a close contender in these regards, and has the added advantage of using less power than a Bluetooth network, but is less capable of high speed data transfers. As an input method, a GUI on the network hub will be needed for system configuration, and voice control will be implemented as the user control input for the system. The DTW based voice control input can be achieved via a simple and small Bluetooth microphone connected to the hub. This input method will require the user to have the ability to make consistent noises and remember the function of each sound. This hands free capability is an input method that can be used by a large selection of the systems target users.

# Initial Methodology

## Overview

The aim of this experimental thesis is to design and implement a voice controlled automation system for a disabled person’s home. The following chapter outlines the methodology and approach taken at the beginning of the thesis project. This can then be compared to the actual design results in chapter 5 to formulate conclusions about the success or failure of the projects goals and method. The design is to take into consideration factors that affect disabled people such as cost, ease of use and implementation, to make it suitable for the largest target group size. There are four key physical components, and five key virtual components to this system. The physical components are a Bluetooth capable PC to act as the hub or core of the system, a device capable of wireless controlled switching and sensor monitoring, a microphone for voice input and a smart phone. The key virtual components of the system are the voice recognition and network management system to be based on the hub PC, the various communication components on each device forming the Bluetooth based network, the smart phone application that will allow for control of the device, and the software that will run the switching device (node).

The system will utilise a wireless communication method so that it may be installed or changed easily. The Bluetooth protocol will be used to facilitate this because it is a cheap and simply implemented communication standard that is available in many household devices. The core hub based architecture of this system will allow the network to be expanded or decreased as needed. The hub will be used to perform the intensive calculations and monitor the entire system as a whole. Each node is designed to be able to handle its own local requirements and communicate with the hub. This means the nodes will each only need a small amount of computational power, making each node both smaller and cheaper. A PC capable of acting as a hub is a commonly available item in most households.

The system as a whole will allow for power dependant devices throughout the area of instillation to be controlled via voice commands, give basic voice control of a smart phone and have the infrastructure available to add sensor controlled devices to the system.

## Procedure

To design and build this system, various components, software, equipment and techniques will need to be utilised. This section will outline the procedure that will be used to design each part of the system and justify choices about the use of specific components used in the design. Spares of each cheap component will be ordered for this system to reduce the consequences of component failure.

### Wireless Switching Adapter

The Wireless switching adapter will be designed to be able to meet Australian safety standards for that category of device after minimal adaptation. Many of these specifications are outlined in the document AS/ASN 3000 Wiring Rules, with particular note to the testing and tagging section. Designing to these specifications will allow the prototype to be in a state that can be more easily improved into a final state suitable for use in Australian homes. The wireless switching device will form the nodes used throughout this network and in addition to switching, it will include functionality for sensor data monitoring and output simple trigger signals. This will allow extra devices to easily be connected to a node and the overall system. A microprocessor will be needed to handle the logic involved in signal processing and operating the Bluetooth transceiver. With the current advances in microcontrollers, small controllers that have enough memory, small power consumption, adequate inputs and outputs and processor capabilities for this task are common. The decision on what kind of controller to use was based off of controller price, company support, availability of specification data sheets and free samples, available programming methods and reputation for quality. This still left many different devices to choose from, so the Texas Instrument (T.I) MSP430 ultra low power microcontroller was chosen due to familiarity and meeting all other requirements. A wireless Bluetooth RS232 TTL transceiver module was chosen because the device is cheap, it is not power intensive, and is simple to operate. A power supply connection, standard 240V 10A relay, MOSFET, diode and a 10 pin header will also be needed for this device. These items will be used to supply power, switch devices, control the relay, circuit protection and access the processors inputs and outputs respectively. The design will first be made and tested on a bread board to ensure it works before being implemented on a PCB circuit board.

### Network Hub & Automatic Speech Recognition (ASR)

The network hub will be designed to act as the central processor for the network. It will handle the speech input, recognition and actions of the ASR interface. The hubs program will be written in C++ and MATLAB as they are familiar and versatile languages. The software will need to be able to analyse and evaluate key voice phrases, and then transmit actions to be executed to the correct nodes in the network. The software for the hub will be designed to run on windows systems as it is readily available for testing and sold as the standard PC operating system throughout Australia.

### Smart Phone Integration

The smart phone integration will also be achieved using Bluetooth communication. Bluetooth is available as a standard accessory in almost all available models of smart phone on the market. To handle the communication from the phone end, and process and execute the commands sent to the phone, a smart phone application will have to be written. This application will need permission to access the phones contacts, phone numbers, Bluetooth communication, and to make phone calls. The application will be designed specifically for smart phones running the Android operating system as several models of phones running this system are available for testing purposes. The operating system is open source and the development software is free. For this smart phone OS, the applications are coded using the java language. Handling of most common functions of the phone will be achieved by calling and accessing standard supplied API’s for the operating system supplied with the development software. The final goal of the application is to be able to recognise and execute commands sent to the phone from the hub, such as ‘call home’. The secondary task of this application will be to pass voice data to the Bluetooth microphone ear piece connected to the network during a call.

## Experiments

This section of the report contains outlines of the experiments to be undertaken to assess the success of the project design.

### Experimental Scope

For the chosen design there are several design characteristics that must be considered and tested to ensure the design will be practical, not just in theory but in practice. These include, the switching capabilities of the design, accurate communication throughout the network, control of the smart phone and accurate ASR. If any of these design characteristics do not behave as expected the design will require changing. These characteristics however cannot be tested without building a prototype model of the network.

### ASR Voice Recognition Experiment & programming

The DTW method of speech recognition will be used for this system. To test that it is functioning properly, first a voice input must be sent to, received, saved, and replayed by the hub computer. This will ensure that the voice data is being received correctly by the hub PC. After this has been achieved, two identical sound files should be passed through the detection algorithm so see if a match occurs. Then one of the samples should be edited to be a faster audio file, and the test repeated. Then the frequency of the sound file should be uniformly increased and then tested again. This will give an indication of the limits of the detection algorithm. Finally, when all these tests have been sufficiently passed, the algorithm should be tested with repeated real voice inputs.

### Hardware Switching Experiment

The goal of the hardware switching experiment is to test that the nodes designed for the network are able to switch a power source when given the appropriate command. To test this, the initial design will first be made on a breadboard, so changes to design and be made in the future. The microcontroller in the wireless switching device will be coded first to simply toggle between sending a high or low signal down one of its outputs to the MOSFET. This should trigger the MOSFET to open or close its gate, allowing power to flow through and in turn trigger the relay. The relay will have a low voltage, battery powered current running through it, powering an LED. If the circuit works correctly, the LED will toggle as the signal supplied to the MOSFET toggles. If it does not work, the voltage applied to the MOSFET’s gate, the voltage on either side on the MOSFET’s channels and the voltage supplied to the LED on either side of the relay can be checked using a multi-meter. These voltages can be checked against their expected voltages to trace where the fault occurred.

Important Notes:

1. All power switching will be tested using LED lights powered by low power dc batteries to reduce the risk of potential harm via electricity. This will act as a proof of concept that the switching mechanism is functioning.
2. The device will never be tested with a live power source unless by a trained and qualified professional.

### Sensor Input & Bluetooth Communications

The aim of this experiment is to determine that the sensor input channel on the wireless switching node and the Bluetooth communications between the network hub and node are functioning. The hub will be used to initialise the Bluetooth connection, and the node initially will be designed to respond to all requests and have no security. To do this, the microcontroller will be coded to regularly check the state of its received (Rx) communications line. If data is being received, it is to check it against a reference stored in memory. If there is a match, the microcontroller is to start toggling the switching signal used in Section 4.3 of this report. This will show that wireless switching commands can be sent from the hub. The second part of this test is to transfer received data from an input pin, to the transmission line connected to the Bluetooth transceiver. This will show that the node can communicate with the hub and that the input signal on the node is functioning. To test this, a known signal will be applied to the input pin of the microcontroller. The controller will be programmed to check on a regular cycle if there has been any input and if there has, then to store it in a buffer. The microcontroller will then send the buffered data down its transmission (Tx) line to the Bluetooth transceiver which will broadcast it to the networked hub. The voltages of transmission line can be checked to make sure that the data is being sent to the Bluetooth module. Testing that the data is actually being broadcast is more difficult as a fault could occur in the design or coding on either end of the transmission. With the lack of a Bluetooth ‘sniffing’ device or software, the code on both ends of will need to be reviewed and tested until a successful transmission occurs.

### Smart Phone Control

This experiment could have been simulated on the computer using the application design software and would not have required constant transferring onto the phone to test the application. The application for the phone could have been coded to ask on instillation for access to full Bluetooth control, contacts control, and permission to make phone calls. The application could have been programmed to run as a background process and to check for incoming Bluetooth signals. When a signal was received, it would have evaluated to see if it was for that application. If it was, then it would have executed the commands contained in the received data. The phone application section of the report was not completed due to time limitations.

## Progress (at time of proposal completion)

Preliminary work was made to design basic hardware designs of wireless communication and switching of electronic devices during the proposal stage of the design. This was necessary as without an outline of design choices, components could not be ordered. Orders were made and received for parts all initial designs had in common. Parts required for the design that had not yet been received in this manner were readily available from local electronics shops. An extended quantity was obtained for all items that required ordering to mitigate the risk of items being broken during testing or not functioning on arrival. Likewise slight variations of the processor were ordered in the event that one model did not fit the design as expected. The software to code and emulate applications for the android operating system had been installed and three basic test applications were been made but further phone programming took place. Finally a small MATLAB application capable of importing an external audio source, sampling and filtering it had been written.

## Risk Assessment

Attached at the end of this document as Appendix A. The risk assessment document is standard practice when commencing most large tasks in industry professions, laboratories and jobs where unexpected dangers may occur. The purpose of this document is to make the user stop and think in advance of all the possible steps that are to be undertaken and how they will be achieved. Then identify possible risks or hazards that may occur for each step, evaluating the total risk for harm. Finally control methods are put in place for each possible cause of harm to reduce the risk, and the total risk is re-assessed. There are five categories of control methods that are commonly referred to, these are elimination, substitution, engineering controls, administrative controls and personal protection equipment (PPE). If the risk is above a certain threshold even with all the control measures in place, then the task is too dangerous to undertake.

# Design Results

## Pc Side of Project

The PC side of the project was created using MATLAB as the programing language due to its familiarity and many functions for mathematical operations. The functions allowed for easier implementation of the DTW needed for the speech recognition of this project. The first task was to break the project into sections based on functionality, and treat each of these as design steps and individual goals.

### Audio File Handling

The first task was to create a method for opening, reading, storing and writing audio files. This was a critical task because without this functionality there was nothing to use speech recognition on, nothing available to test and no conditions to trigger commands to send to the microcontroller on the switching node. Wave files were used for this application because functions [[26](#_ENREF_26)] to sample and manipulate a wave file come built into MATLAB’s standard functions and the .wav file format is easy to manipulate. Functions for manipulating other audio types exist and are freely available online [[27](#_ENREF_27)], but the accuracy and efficiency and hence reliability of these functions are unknown. The functions referred to [[27](#_ENREF_27)] does have significant documentation and descriptions of techniques making them useful for forming an understanding of how similar projects are implemented. This is not the case for many other third party solutions which often have little documentation. Several freely available .wav files were collected from the internet to use for testing. For testing purposes, a GUI based file select was implemented for selecting the wav files as seen in Figure 1.

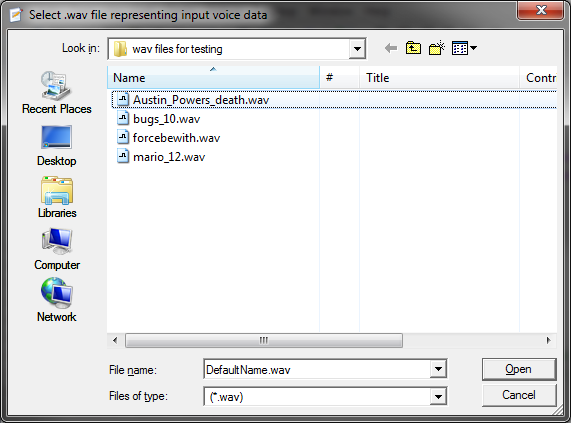


Figure 1– Wav file selecting Graphical User Interface

These files were then sampled and the values written into arrays. These arrays of values were then ready for later DTW testing. Initially the information from all audio files that had been read were stored in the same array. This was to reduce the number of different variables used. This caused problems when DTW was implemented because each audio file was not of the same length. At similar sampling rates each file could have different number of total samples which is a problem in an array because each element of on array must be of equal sizes. Rather than store the data in an array the length of the longest audio file and have values of zero at the ends of shorter audio files, the data for each file was stored separately. Once this functionality was proven to be working, the code was used to read from a set directory and build variables that contained the data from each .wav file in that folder, see Figure 2. When speech recognition was implemented the mkdir function was used to create a location to store each of the .wav command files. These files are re-read when the program is restarted and then checked against for recognition when voice data occurs.

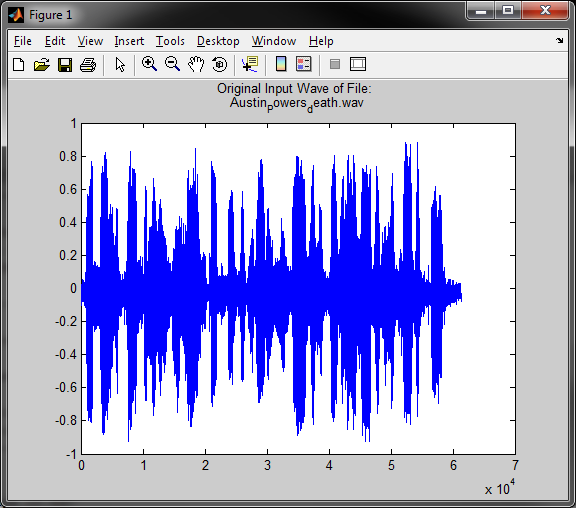


Figure 2 – Austain Powers voice data plotted from the sampled data

One problem that was encountered during audio file manipulations was that the MATLAB functions often utilised system codecs for decoding/encoding the audio files. If These functions are used on a PC that does not have the required codecs installed the MATLAB functions will cause errors and crash the program. Newer versions of the Windows operating system comes with many standard codec functionality installed as standard and the .wav file format is not recent nor an obscure file type, but correct operation of these functions could not be guaranteed.

### Pc Comms Port Monitoring and Communications

A comms port is communications port on a PC. Initially comms ports were always a physical port on a PC such as the RS232 serial port. These ports were built onto the motherboard of a PC and always expected to be there. They were often assigned a comm port number, and it was never expected this number might have to change or that it might come into conflicts with another port using the same name. This is no longer entirely true, virtual comm ports can now be created and utilised over usb, Bluetooth or even wireless communication protocols. In the case of this project, the USB connection to the Launchpad development board was first used to test communications. Once these communications worked properly the project altered to use the Bluetooth communications. The transition required little more than setting up the Bluetooth connection as a serial connection and changing the port used because the type of communication did not change from serial communications. To complete this implementation this section of the program was further broken down into several smaller goals. These were to develop a series of commands and command structures used for communicating with the microprocessor, a serial type terminal interface to show what has been sent and received, and the code to actually monitor a particular com port for communications and direct that information to where it was needed.

Before any of these steps were completed on the PC, the microcontroller had to be programed and tested to successfully transmit and receive information. Otherwise it would not be possible to know if an error was caused on the PC side of the project of the switching node side of the project. But this also required a terminal like functionality to test the microcontroller. For this reason the program ‘PuTTY’ was used. PuTTY is a terminal program capable of communicating using several different communication methods, one of which is serial, see Figure 3.

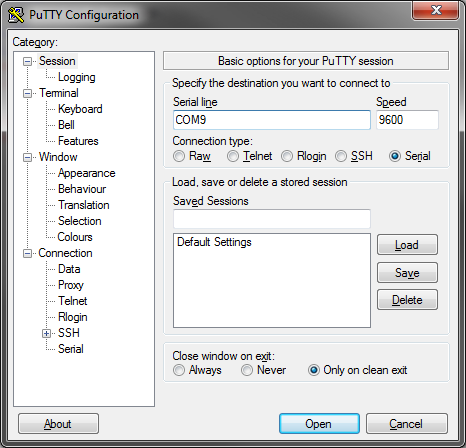


Figure 3 – PuTTY terminal configuration screen, connecting to COM9

Once communications were functioning correctly from the microcontroller and expected known data and responses were setup, work began on implementing a MATLAB program specific interface rather than use the terminal. Initially the microcontroller was programed to loop forever sending an incrementing number, pausing, and then sending another. The MATLAB fopen, fscanf, fprintf and fclose functions were used to open, read from, write to and close the comm port used for communications. Combined with the correct settings for baud rate and number of stop bits the data was easily sent and received. Once this was completed, the first version of a program specific interface was created for the project. This was created using basic string, display and variable operations, see Figure 4.

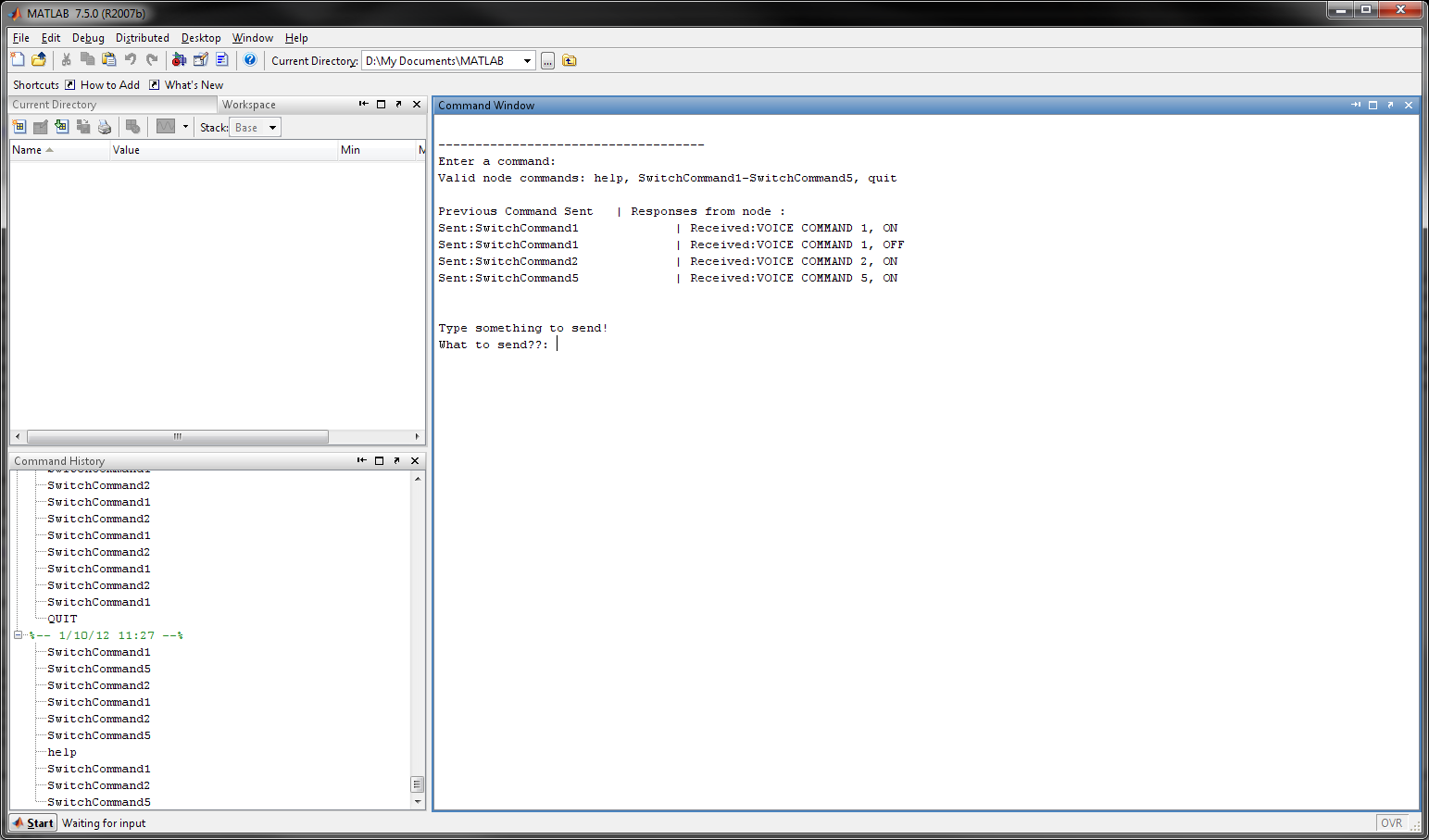


Figure 4 – First version of comm interface

The interface displayed valid commands as well as record the command sent PC and the response from the microcontroller. The help command triggered the controller to transmit the same information as given at the top of the interface, SwitchCommand1 – SwitchCommand5 triggered the controller to pull high or low the output of one of five I/O pins and respond back to the PC if that pin was ‘ON’ or ‘OFF’.

### Dynamic Time Warping

The dynamic time warping was used in this program to find the similarity between two different lots of speech data. As new voice data was collected it was broken into lengths that matched the same length as command files. Command files were the voice data stored on the PC in a specific directory which all voice data was compared to for recognition. The DTW used in this program followed the detailed online explanation and example shown on the webpage for the Laboratory for the Recognition and Organization of Speech and Audio (LabROSA) in Columbia University New York [[27](#_ENREF_27)].

The DTW handling function for this software only accepts five arrays of data from command files and the sampling rate associated with each lot of data as well as the voice data that was to be compared to the commands. The function repeats the same operation five times between each of the five commands and the voice data. Fist the function would find shortest of the 2 files being compared. This was not necessary for commands recorded using the command input function as the length of these recordings are the same as the length of the samples of inputted voice data. However the software allowed for users to select any .wav file when the voice input test function or the command setting function was used. The DTW handling function would then read that much of the longest file from the beginning of the file. This made the two amounts of data samples equal for comparison purposes. Rather than taking that length of data from the start of the longest file on attempts were made to take the data at even increments from the entire length of the longest file to make it match the size of the smallest file. This had poor results and was a more resource consuming process so it was not used.

The MATLAB spectrogram function was then used on each set of command data and the inputted voice data. The spectrogram function returns the spectrogram of a signal, in this case the command and voice data, into matrices. This function used a short-time Fourier operation and must be passed the array of data, the length of segments of the data, the number of overlaps, the number of frequency points to use to calculate the DFTT and the sampling frequency. The number of frequency points that were used was the functions default of 256. Each column of each matrix contained an estimate of the short-term time-localised frequency content of the command or voice signal passed to it. Time is represented across the columns from left to right and frequency increased down the rows from top to bottom.

Then the Simmx function was used being passed the absolute values of the spectrograms of the voice data and each of the commands in turn. This function used the two matrices to construct a ‘local-match’ scores matrix of the distance between the magnitudes of each of the matrices. Initially before the use of the simmx function, difference distances were calculated in vertical and horizontal values rather than the diagonal cosine distance used in the simmx function. The information was then plotted and the dynamic programming function [[27](#_ENREF_27)] was used to find the lowest cost path (smallest total difference) from corner to corner of the cost matrix (Figure 5). The lowest cost path was then plotted over the top of the other matrices. The shorter in length of the difference path, the greater the match was between the two compared files.

This value was then used in a conditional logic statement where if the total difference was below a set threshold then the voice data was considered a match and a specific action was taken in response.

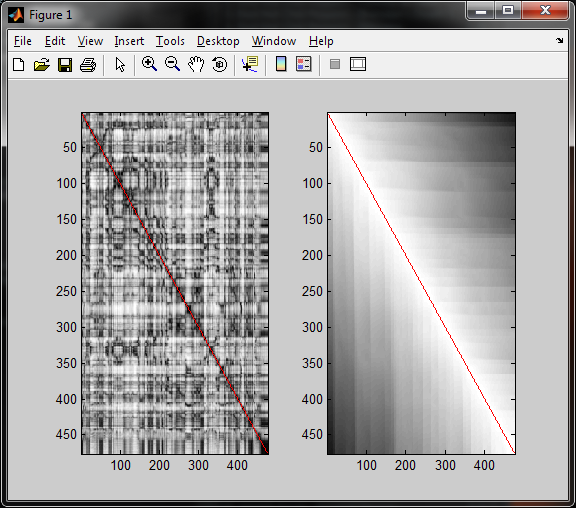


Figure 5 – Lowest cost path of two identical waveforms

### Audio Capture

The audio capture software was necessary for voice input and was one of the most problematic parts of the design. This required pairing a Bluetooth microphone to the host Pc and then implementing MATLAB code to monitor and collect data from the associated comms port. Using coding from the previously completed audio handling section of the software, the voice sample was either stored, if it was a command being set, or checked against the data base of commands if it was inputted voice data. An obstacle that was not designed for during the initial phase of this project was a method to define if the audio input was for setting a new command or checking against currently stored commands for recognition. This required some extra unforeseen developing before further functionality could be implemented. The simplest method was to have the user set a flag, when setting the voice data for a new command. This was achieved by having an option in the menu system that would set this flag.

Any voice data that was captured when this flag was set was stored as a new command. If the flag had not been set then the captured voice data was checked against the command data database for a match. The flag once set was only set for a set period of a maximum of 10 seconds or until the menu option to un-set it was selected. This method was later replaced with two separate but similar functions, one to handle recording and storing command data and a second function to constantly check for voice input and compare it to the command data. This command setting function was designed to contain a play back and confirmation option for any newly inputted voice commands prior to the data being saved and stored. The user also has the ability to choose the file name for the command. The microphone used for the initial design was a Jabra BT2045 as seen in Figure 6, and was chosen as it used standard Bluetooth protocols and methods to communicate. By designing the software to function with this standard, the largest range of devices should have been compatible. No other brand of Bluetooth headset was available for testing purposes.



Figure 6 – Bluetooth headset used for audio command capture

### User Interface and Debugging

The user interface for this design was a progressive work and the last part of the software to be completed. This is because it was judged the least important part of the software in terms of additional features and design. A well designed, easy to use interface is very important for an end product, but this design was for testing a proof of concept. Initially, as each previously mentioned section of software was written and tested, the results were printed to the command window of MATLAB rather than use a traditional GUI. Then rather than output and print the results of the variables to the screen, additional information was added. This information contained a basic description to each of the data outputted and of the purpose of the program.

As more functions and separate parts of the software and functions were completed an option system was added to the interface. Selecting different options caused different actions to be taken. Additional information about the use of each of these options was finally added to make the menu system more user friendly, see Figure 7. This design, while not as astatic as a traditional GUI allowed for an interface that could easily and quickly be modified as needed and had the added benefit that the MATLAB variable viewer was directly above the command window which was useful for debugging.

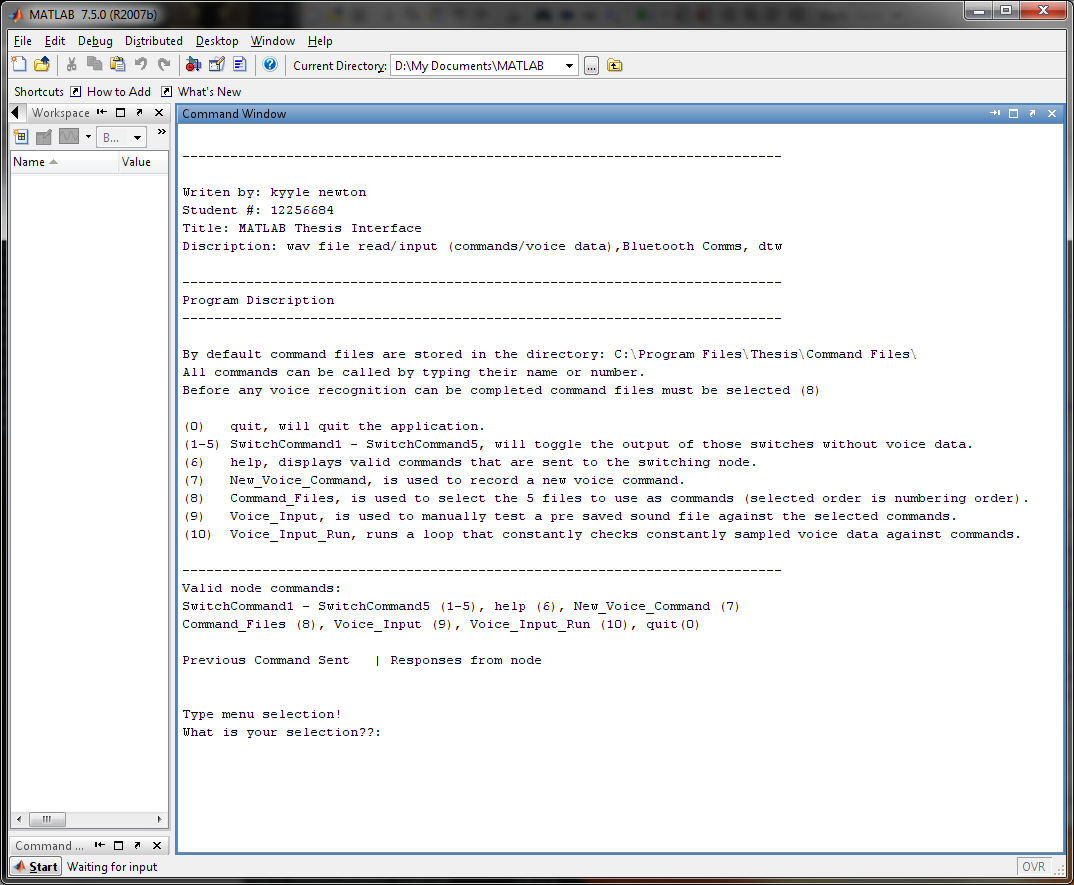


Figure 7 – User interface

The final user interface was designed to be as simple as possible and connect to only one node. There were several menu options in this interface, SwitchCommand1 - SwitchCommand5 (1-5), help (6), New\_Voice\_Command (7), Command\_Files (8), Voice\_Input (9), Voice\_Input\_Run (10) and quit(0). The CommandSwitch1-5 options made the node switch ports 2.1 to 2.5 respectively. The New\_Voice\_Command command was used to record new voice commands. The Command\_Files option was used to select and read the previously saved command files into arrays to be used for recognition. The Voice\_Input option allowed for pre-recorded voice data to be checked against the command data for recognition. This was used for testing and allowed the recognition option to be run at will with edited and saved .wav files, not just during user voice input. The significance of this is that it allowed voice data with known properties to be evaluated to observe the results of the DTW. These results can be seen in chapter 6 of this document. Finally the Voice\_Input\_Run command was used when the program had already loaded command data. This option put the program into a state where the sounds from the Bluetooth headset were constantly recorded and checked for recognition against the command files. These menu options were used to test switching and the DTW speech recognition was working correctly as well as for the programs intended operation.

### Complete PC Side Operation

A brief outline of the flow of operation of the PC software:

1. Assignment and opening of comms port for communications with the switching node.
2. Assignment and opening of comms port for headset audio communications.
3. Display the menu system, options and information.
4. Check the user input.
   1. If its voice data input, check it against command data for a match.
      1. On success send appropriate command to node.
5. If user input is something to transmit to the node.
6. Transmit it to the node.
7. Check for a response.
8. Node reports errors or confirmation of operation and switch states
9. If the command is PC side only.
   1. Set flag for next headset audio to be saved as a command.
      1. Headset sends audio data.
      2. Play back recorded data.
      3. Check for confirmation.
      4. Archive captured audio.
      5. Request the user repeat sound as test afterwards
   2. Displays help information for the program.
   3. Run DTW using a user selected file against command data for a match.
      1. On success send appropriate command to node.
   4. Quit the program

## Node Side of Project

The second key part of the project was that of the wireless switching node. This included elements of both hardware design and software coding for the microcontroller. Like with the PC side of the project, the design and implementation of the wireless switching node was broken into smaller individual goals and tasks to enable efficient design and resource management.

### Components

The following section of report outlines significant components of the wireless switching node. This includes both the hardware of the node (Figure 8) as well as the software of the node. This section of report will further elaborate on how the design was implemented and the characteristics of each component.

#### Node Diagrams

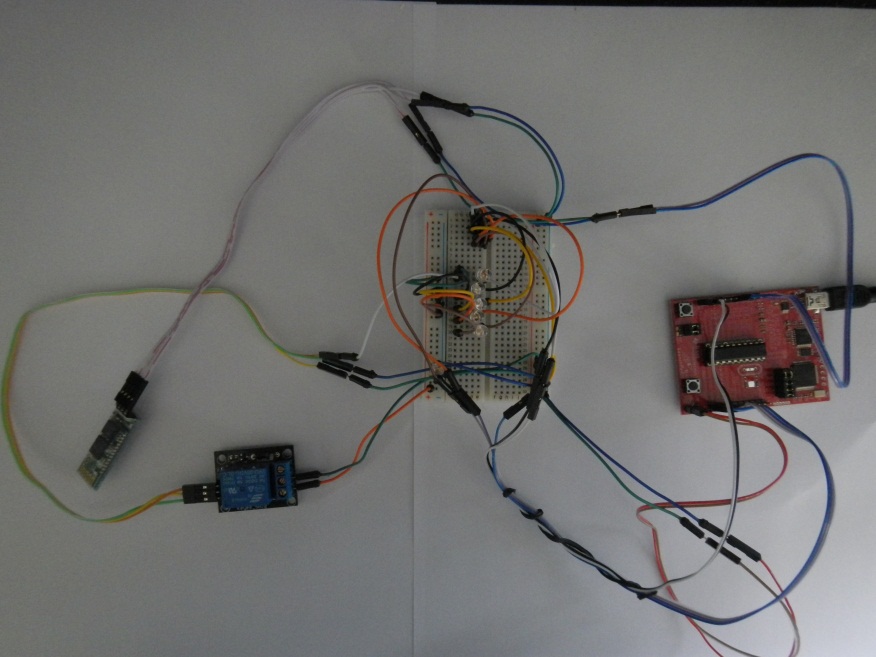


Figure 8 – Wireless switching node bread board prototype

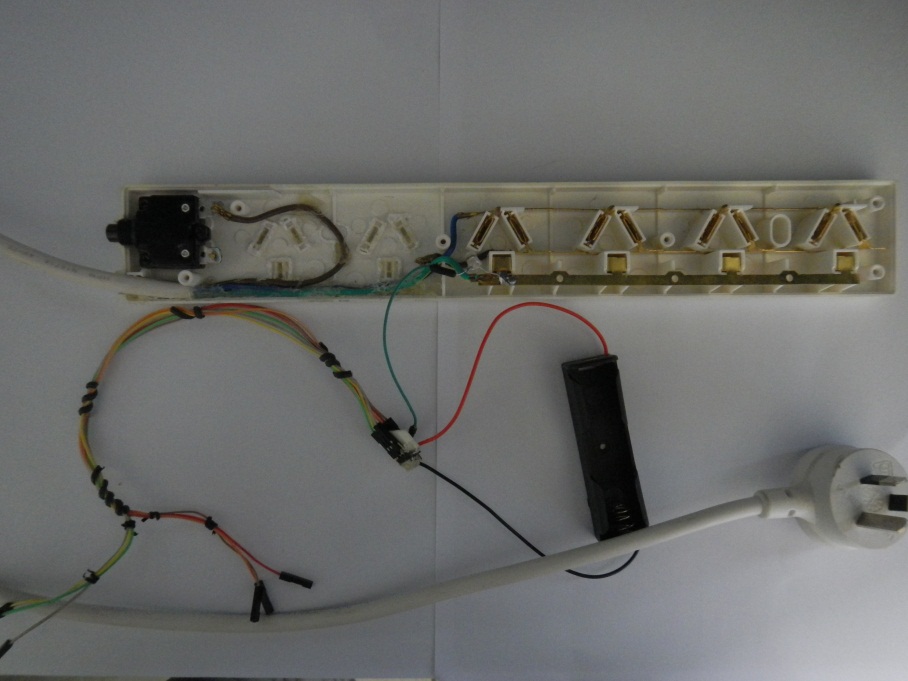


Figure 9 – Power board implementation

#### Microcontroller (MPS430G2553)

The microcontroller used for this project is the MSP430G2553 of the Texas Instruments Launchpad development family as shown in figure 10 with the Launchpad development board.

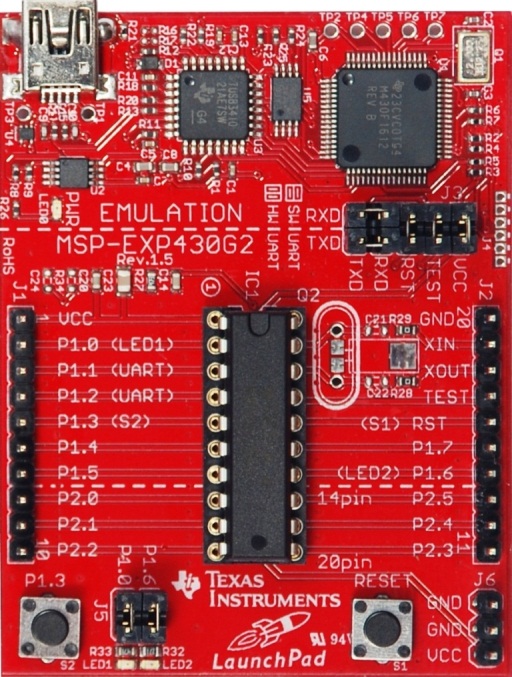


Figure 10 – T.I Launchpad & MSP430g2553

This chip has more than the initially estimated memory, I/O ports and supported communications methods required for implementing the design of this project (Figure 11). Only a single transmit (Tx) and receive (Rx) line, and one I/O port on the chip where needed for this design. The Tx and Rx pins were used for communicating with the Bluetooth transceiver and the I/O pin was used to switch power to activate the relay.

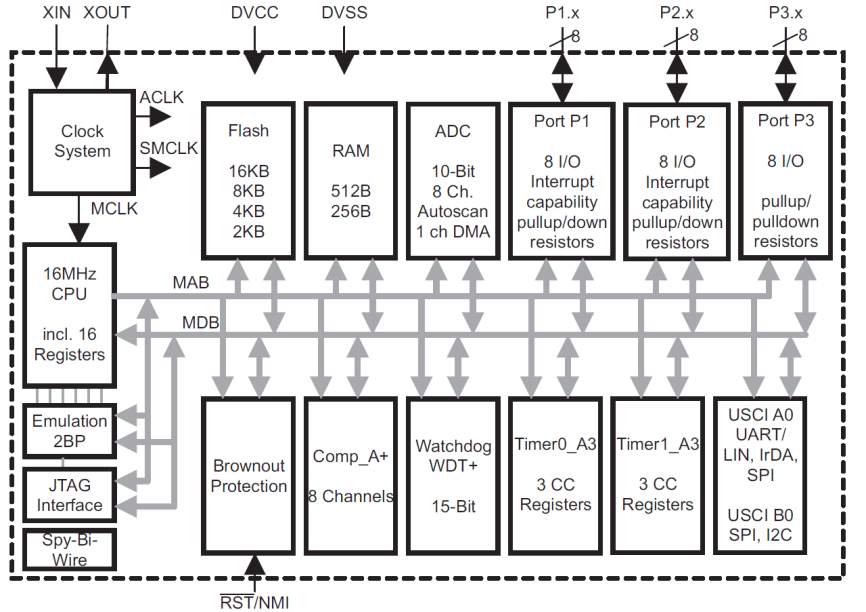


Figure 11 – MSP430g2553 Block Diagram

There are simple many example code documents utilising and describing each of the processor families’ main functions available on the Texas Instruments website. The development board for the MSP430 series of chips has easily accessible headers for connecting to each of the microcontrollers’ pins, 2 LEDs that were used for debugging purposes and on board hardware based debugging chips. The board also contained USB, and functionality for breakout boards to be designed and connected over the board directly to the header connections.

The development board, including USB cable and a single microcontroller came to a total cost of $5, with additional microprocessors costing $1-$2 each. The chip was suited to controlling small operations and being used as the primary processor for communications with remote or remote controlled designs. Additionally there was a large and comprehensive amount of documentation describing the functions of the board and how to configure the registers for specific operating characteristics.

Deciding which of the microcontrollers’ functions would be utilised for the design and what settings would be required in their use was the first task approached for the node side of the project. Normally the first step would have been to consider the hardware characteristics such as available flash memory, processor speed and ram to choose suitable hardware. This part of the design was completed during the proposal stage and this particular chip was chosen because it had significantly more memory and processing power then was required at a low cost. Deciding what controller functionality would be utilised was important because some of the functionality of the microcontroller was exclusive of other functions. This means that the operation of the microcontroller had to be carefully considered and designed to only use abilities of the controller that would not be in conflict with other required functions (on board timer circuits, communications methods and I/O).

All coding for the microcontroller used in this project was completed using a free trial version of Code Composer Studio (CCS) (Figure 12). The trial version of this software allowed for programing MSP430 chips with a memory of up to 16kB, which was the amount of memory on the MSP430g2553 chip.

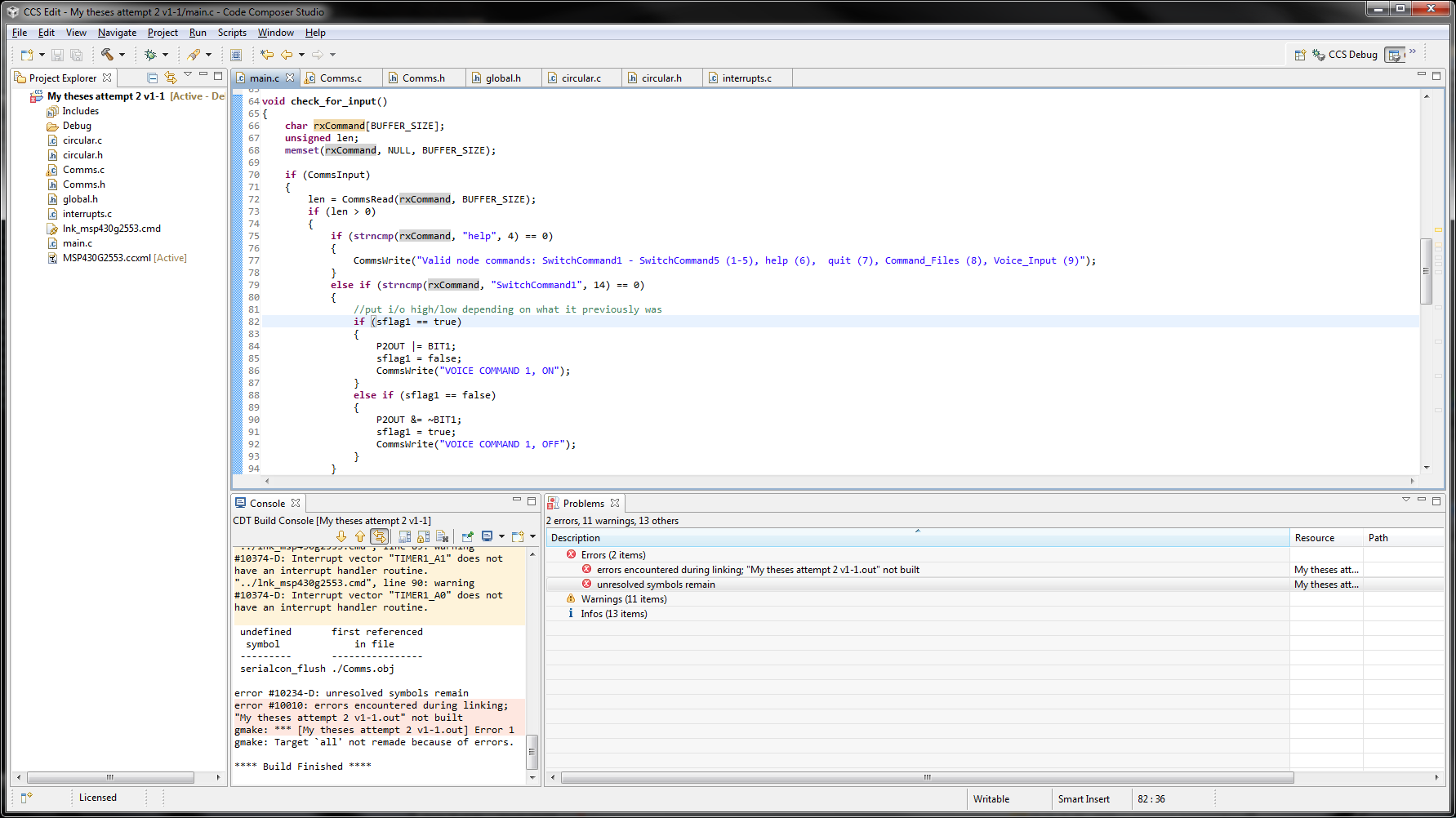


Figure 12 – CCS programing interface

Once the required controller functions were determined, the next step was to configure and then initialise these functions. Depending on the functionality and model of the controller being used, this could have had included setting some pins to transmit or receive data of an expected type, for example when utilising the hardware based UART on the chip. This made those pins no longer available to be used as standard I/O pins. Another common initialisation process is the initialisation of counters or timers used to control various processes. Other types of initialisation required at the beginning of the controllers program may have included assigning sections of memory of set sizes for specific operations, so that its memory addresses are always known and will always have enough memory to operate correctly.

Some of the initialised variables and registers for functions were set using registers that are read when the controller first powered up. These were the most important controller settings and are read before any other programming operation occurs. These registers are bits that are assigned to specific locations in the controller’s memory. The location of these registers always remains the same, and as the device is powered, it will always expect these registers to be at the addresses the compiler has told the controller they are located in. Each single bit refers to a single setting, with settings that are related or exclusive of each other grouped together in one byte groups. These groups are called the registers, and specific bits of each register are set to a binary value of ‘1’ or ‘0’ depending on the desired option and the documentation for the controllers design. In this project registers had to be set to show which pins will be used. For example, if the pin was used to transmit or receive, or to have se the output of the pin high or low. It is also possible to set interrupts for when something is received on a pin. To save power pins that are not used were declared that they are not in use using the registers.

The second type of initialisation used before any of the rest of the program was run by the controller was the creation of variables that were used throughout the program, setting required amounts of memory, initial variable values and assigning clocks and counters of the controller to specific tasks.

The settings and values of the registers were configured one bit at a time using the C language bit wise logical operators (and, or, xor, ect) or all at once. To use this method, the current register value and there the current eight settings stored in that register have a variable assigned to point to the memory location of the register in question. From that point on, any operation on the bits of the variable were operations on the bits of that register as well, as they were both pointing to the same location. The simplest way found during this project to change a single value without changing any existing settings was to use the OR ( | ) or AND (&) bitwise operators. The OR operator compared the bits in two different bytes and resulted in a 1 if either of the compared bit values contained a 1.

Bit wise OR example:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 |
| Register: | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| Comparison Value: | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Result: | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |

If the bit position was a 1 in the register OR the comparison value, then a one is placed in the result at the same bit location. This allows bits to be changed to a value of one while keeping every setting that was previously on at the same value.

Bit wise AND example:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 |
| Register: | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| Comparison Value: | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| Result: | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

If the bit position was a 1 in the register AND the comparison value, then a one was placed in the result at the same bit location. This allowed bits to be changed to a value of zero while keeping bits that did not require changing as their previous settings.

The other method for assigning values to the register was to actually assign the value that represented all the settings required, every time the value of the register was changed. This was not used in logical operation in the controllers programing but was used during initialisation. In the previous OR example, the value of the register was 157 (decimal), to get the same result as was achieved in that example without using the OR operator, a value of 221 (decimal) would have had to be assigned to the variable pointing to the registers memory address.

Once all the registers were set up to use the desired controller functions with the required settings and all variables that are required throughout the program are defined and initialised, the actual program itself needed to be written.

Interrupts are an action that occurs only if certain pre-defined conditions are met. These can be based on a specific action or event happening such as a flag being set, or on a regular interval defined by a clock pulse. Interrupts were able to cause complications because some interrupts with higher priorities were able to interrupt an already running interrupt which made checking for logic errors much more difficult. For this reason they were used as little as possible during the design of the project.

The only interrupt that was used in this project was triggered by the condition that information has been received on the Rx pin. If this was true, the controller would break out of what it was currently doing and read each byte of data that is received into a pre made circular data buffer for later use. In later designs this was no longer entirely true, a function was written for checking if anything had been received and placed in a loop that would run the function periodically (figure 13). This change was made because after the initialisation was completed if nothing was received the controller was just delaying and waiting for an interrupt anyway. This also gave a greater level of control of the communications from the controller as other transmit functions could be added to this loop for debugging. The interrupt still occurred and placed any received information in the receive buffer but that was the limit of actions that occurred during the interrupt. The function would then check if there was anything in the buffer.

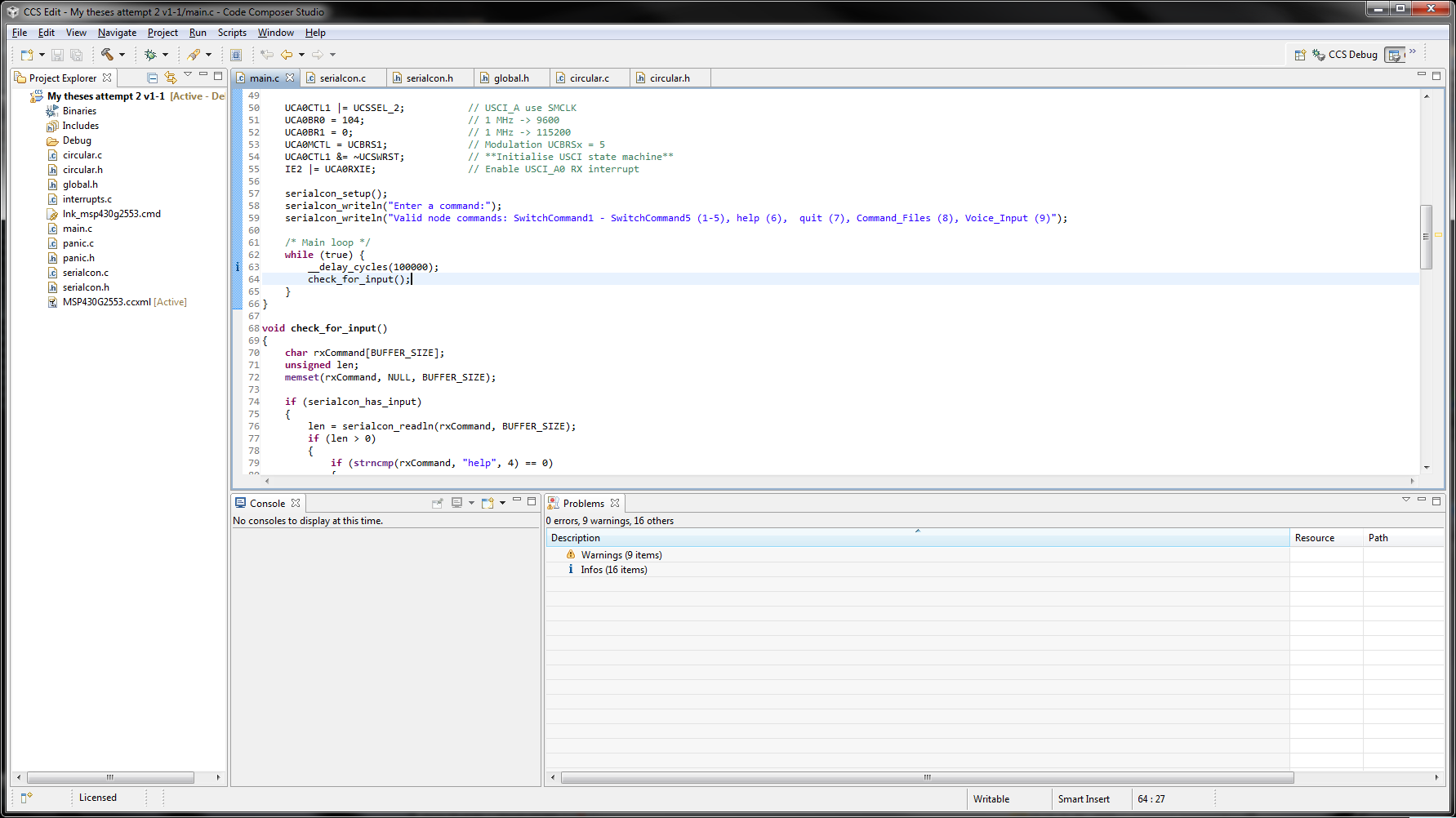


Figure 13 – Microcontroller main loop after initialisation

The buffers for the microcontroller are used for storing data when it is received or prior to being sent. These buffers are implemented by creating a structure all buffers will follow. This struct is then used to create each buffer. The format of these ‘BufferType’ structs is an array of char elements, an index to the current read position in the buffer and an index to the current write position in the buffer. Once this structure was created, functions to clear the buffer, write data to a buffer, read data from the buffer, check if the buffer is empty and check how much data is currently in the buffer were written. These functions are used for any basic operations on the buffer, utilising the array structure to store characters and the indexes to keep track of write and read positions (see appendix). The functions for the buffers were implemented in a circular nature, so that if the function writing to the buffer hit the end of the buffer, it could write over information at the start of the buffer providing that the information had already been read. Buffers that work in this fashion are known as circular buffers.

The outline of the operation of the node is as follows:

1. Initialise registers, clocks, timers, pin masks, buffers, interrupts and variables.
2. Wait in a loop.
3. If something is received, generate an interrupt.
   1. Read data as it comes in from the Rx pin into the Rx buffer.
   2. When all data is received, compare it to a switches case arguments.
   3. If it matches a case, complete that code.
      1. Switch the power to a device by toggling the voltage on a I/O pin.
      2. Fill Txbuffer with a message saying a correct message was received and that the operation was completed.
      3. Send the message.
      4. Clear Txbuffer, Rxbuffer and the interrupt flag.
   4. If it did not match an expected case.
      1. Fill Txbuffer with a message informing the PC of this.
      2. Send the message.
      3. Clear Txbuffer, Rxbuffer and the interrupt flag.
4. Leave interrupt and continue looping waiting for another command.

#### Bluetooth Module

The Bluetooth transceiver module used to add Bluetooth capability to the wireless switching node is an JY-MCU as pictured bellow.



KEY

VCC

GND

Tx

Rx

STATE

Figure 14 - JY-MCU Bluetooth Module

The device only provides a four pin header as a connection which is unusual for devices of this type. The connections provided are the power and ground for the device, as well as the transmit (Tx) and receive (Rx) header pins. These are the most fundamental connections required for utilising communication. There were other connections that could be used to format the device, change its operating modes and reset the device that this board did not supply easy access to (Figure 14). Connection points that do not have a header on this module are the ‘state’ and ‘en’ or ‘key’ connections on the chip. The state connection is used to signal if the device has received an AT configuration command or is working in its regular mode. AT configuration commands are a stand set of commands used to configure a variety of electronics.

If the voltage on the state pin in high the device is receiving AT commands and is being configured. If the state pin is low or floating, then the device is operating in its normal functionality. The ‘en’ pin is used only if the Bluetooth device is operating in master mode, which is not the case for this project. If the device is operating in master mode, then only the last slave device to connect to the master is able to reconnect, unless the ‘en’ pin is triggered by a high voltage. These limitations were circumvented by transmitting settings commands to the device down the Rx line and testing with the PC what settings the device reports. The devices hardware is configured by the manufacturer to read messages coming in on the Rx line, and if the leading values of the message are of a set expected format, then the devices accepts them as configuration commands. These commands were used to setup the module for communications between the controller and the pc.

By default the module contained the settings:

|  |  |
| --- | --- |
| Module: | V1.2 |
| Baud Rate: | 9600 |
| Device Name: | Linvor |
| Device Password: | 1234 |

These values are acceptable for the commands that are passed between the PC and controller in this design, as the commands are small and don’t have to arrive at a specific rate. It is also an easy baud rate to configure the microcontroller to operate at, and is a common comm port setting on the PC.

A configuration manual for the HC-06 Linvor Bluetooth chip used in this module provided a list of commands that can be used for configuring the Bluetooth device. The manual also indicated that each command sent to the device may take up to a second to be implemented. As the baud rate for the chip was at a desired rate already, this was not changed but both the device name changing command and pin setting command were used to make the device more easily recognisable and accessible. This was achieved by connecting the device to the microcontroller and programing the controller to send the commands to the Bluetooth device during initialisation. The command to set baud rate was also sent despite not being necessary in this case, this was to ensure that if the Bluetooth module lost its settings or a different module was connected, it would still be configured correctly.

List of JY-MCU Configuration Commands

|  |  |  |
| --- | --- | --- |
| Send: | Receive: | Description: |
| AT | OK | Queries the device and receives a response. |
| AT+(BAUD1-BAUDC) | OK1200 – OK1382400 | Sets the baud rate of the device between one of 12 options |
| AT+NAMEswitching\_node | OKname | Sets the name of the Bluetooth device. Limited to 20 characters |
| AT+PIN0000 | OKsetpin | Sets the password for connecting to the Bluetooth device. |
| AT+VERSION | LinvorV1.2 | Returns the version of the software on the HC-06 Bluetooth chip used on the JY-MCU |

#### Relay

The relay used for this design is a cheap SRD-05VDC-SL-C relay by Songle (Figure 15). This relay was rated to switch up to 10 amps at 250VAC or 10A at 30VDC. The three pins on the right hand side of the picture show from the top, the lower voltage (-), higher voltage (+) and signal (s) connectors. In the case of this project, the “-“ pin was connected to ground, the “+” pin was connected to VCC and the “s” pin was connected to the microcontroller I/O output with a diode on the line for protection.

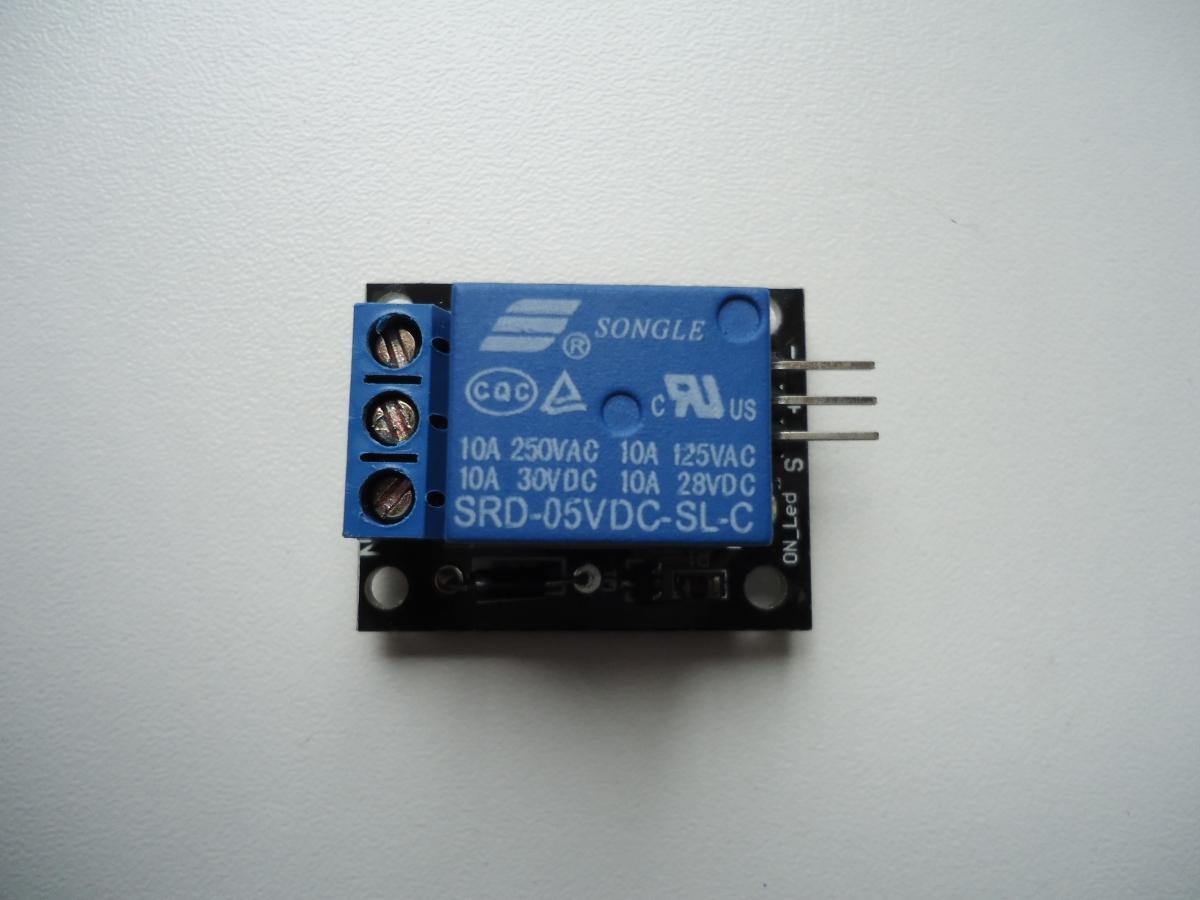


Figure 15 - Songle SRD-05VDC-SL-C relay

As mentioned previously, all design and testing of this project was conducted using strictly low power DC sources. For this reason a single relay was used to test that the node could switch a power source on receipt of wireless commands from a PC using only the 3V used to power the development board. Underneath the relays PCB board there were exposed connections and solder points that can be directly touched. If care was not taken or large amounts of power was used then this could be a source of injury or damage to other components of the node. For this reason, it is recommended that in a final design situation switching higher levels of power that the relay be contained in an isolated enclosure to prevent contact with live components.

Initially detailed specifications for the device could not be sourced, which has been a common problem with obtaining cheap electronic components via the internet for this project. The reported specifications of the relay used in this project were as follows:

List of Relay Specifications

|  |  |
| --- | --- |
| Part Number: | SRD-5VDC-SL-C |
| Brand: | Songle |
| Size: | 19.2 x 15.5 x 15.2 mm |
| Rated Load: | 10 Amps 250 VAC |
| Electrical Life: | 100000 |
| Mechanical Life: | 10000000 |
| Coil Parameters: | 0.36 W coil power, <75% suction and voltage, >10% release the voltage |
| Environmental Temperature: | -40 degrees to ~80 degrees |
| Weight: | 8 g |

From the device part number the voltage required to operate the device was confirmed against the information contained on the website that sold the device. In this case, to operate the device, 5VDC is required to be supplied to the device. This was an over sight during purchase of components, as both the Bluetooth module and microcontroller only require 3.3VDC, which was the voltage the design was based around. This was found not to be a problem, as during testing the device functioned correctly with the some power supplied to the Bluetooth module and the microcontroller (3.3VDC).

On the left hand side of the picture of the relay seen in figure 14, there are three screw down connectors visible. These are labelled from the top, “NO”, “NC” and “C”. These stand for normally open, normally closed, and common. For the design of these nodes a normally open setup was decided upon, this means that there was no need for a connection to the “NC” output. When a signal is provided by the microcontroller a FET will allow power to flow from the ‘+’ terminal to the ‘-’ terminal and the coil inside the relay will energise. The magnetic field from the coil will then mechanically close the switch, enabling power to flow from the common ‘C’ connection and out the normally closed ‘NC’ connection essentially turning a device on.

### Communications

The communications implemented for the node was very simple. The controller using its built in UART, communicated using the transmit pin with the Bluetooth module. The module then received that data on its receive pin on its header, which it then transmitted via Bluetooth to the virtual communication port on the PC and vice versa. The term UART stands for universal asynchronous receiver/transmitter and the msp430g2553 microcontroller used in this project contained a hardware based UART controller. This allowed for the conversion of data from a parallel format to an asynchronous start-stop serial format using an integrated circuit rather than a software based solution. The UART on the controller contained a form of driver that converted the parallel data to a serial format that could be received and transmitted by the Bluetooth module. For these serial communications to have been received correctly, each device had to be paired to be sending and receiving at the correct baud rate. The easiest way to implement this was to have each device communicate at the same baud rate for both transmit and receive. It was also necessary to make sure the Bluetooth device was paired with the PC and connected so that the comms port can be opened and communications can take place (Figure 16).

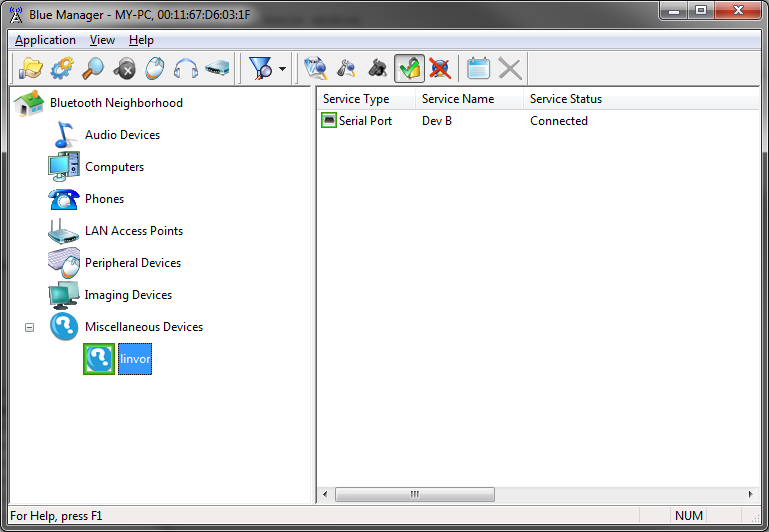


Figure 16 – Blue Manager Bluetooth management suit node connection

Using a case statement on the PC software, a different command was sent over Bluetooth to the microcontroller depending on the outcome of several logical checks (Figure 17) on the PC. The most obvious check was that a voice command match occurred. If a voice input matched a voice command on the PC software, then the PC would send data such as ‘SwitchCommand1’ to the microcontroller.

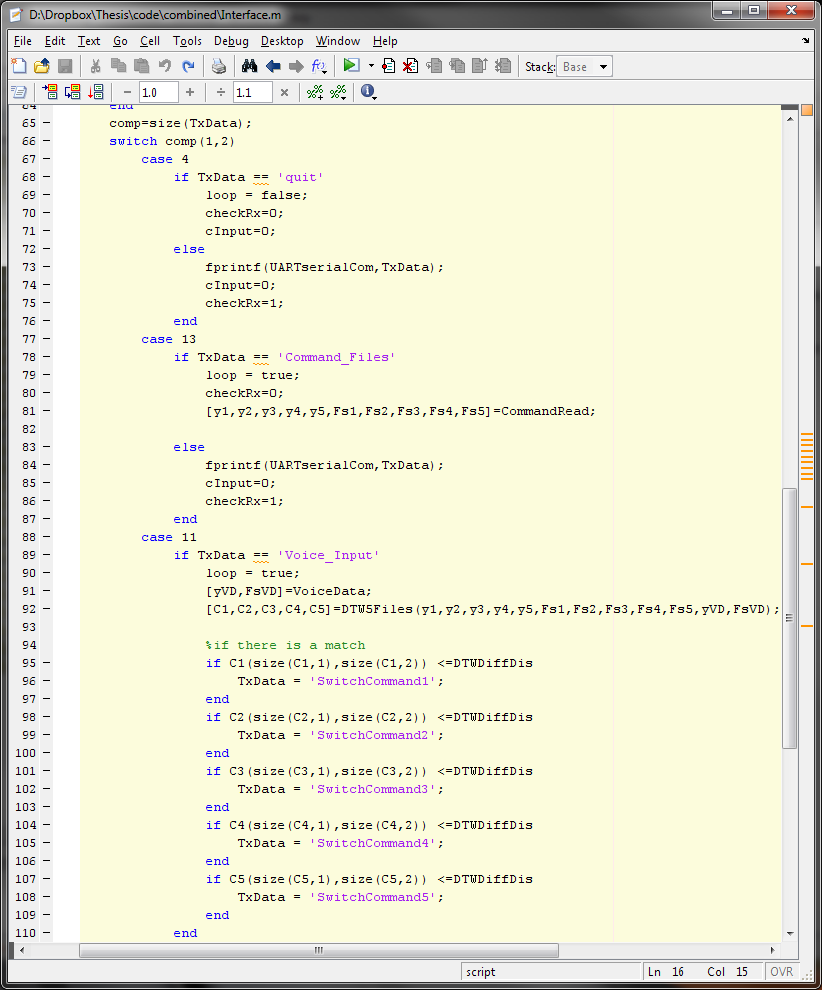
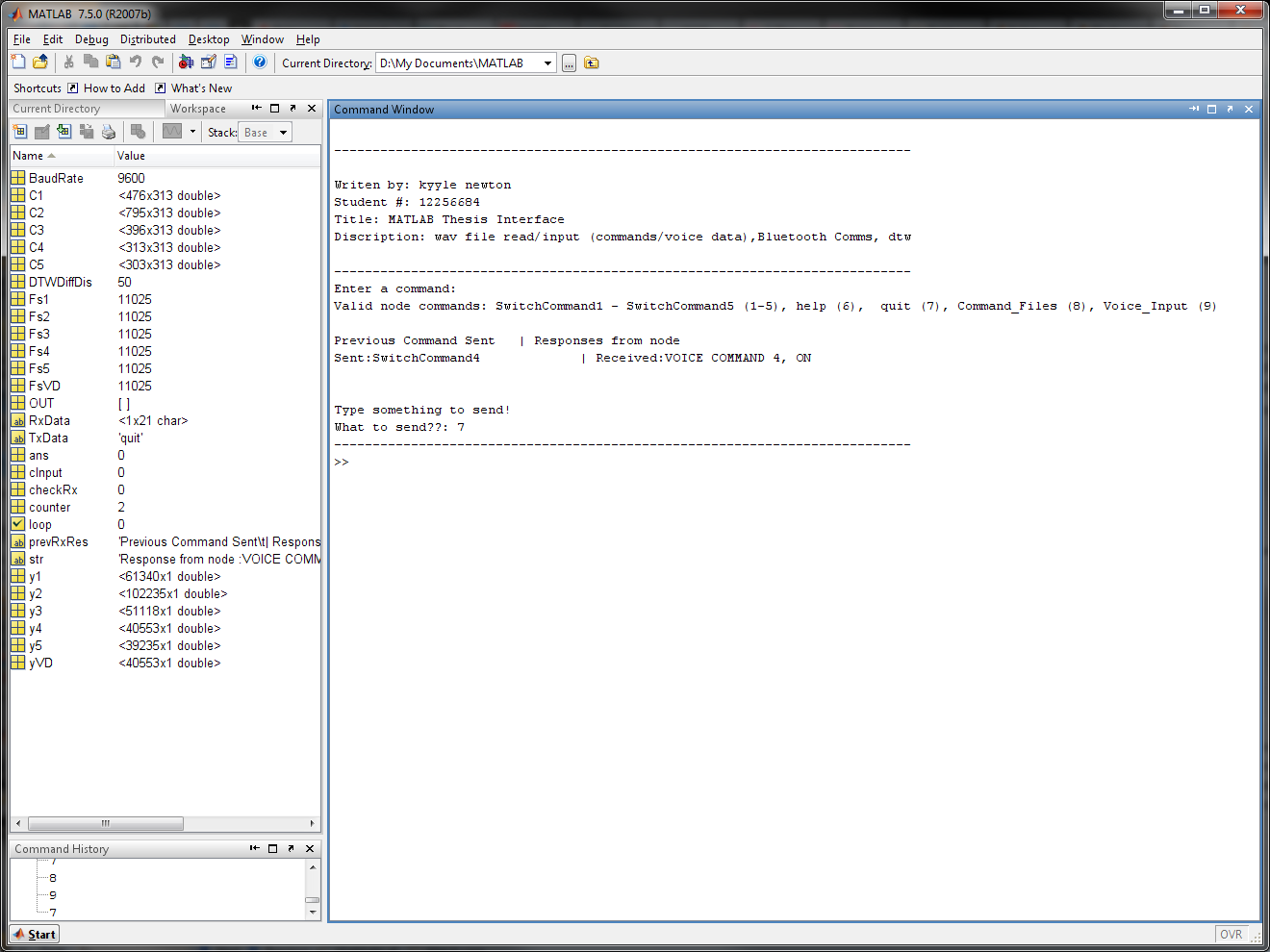


Figure 17 – Excerpt of PC logic checks for communications

The microcontroller on receipt of any data has a receive flag occur which in turn triggers an interrupt service routine (isr). This caused the controller to break out of whatever operation it was currently doing, which in most instances was idling and waiting for a command, and do a set of commands that it completes every time data was received. Each byte as it is received was read into a pre made buffer so that the next byte could be received and no data was lost. Once the termination character had been received for that transmission of data, the microcontroller responded by sending a confirmation to the PC that the data was received, this was primarily for error checking during the initial testing phases but was useful to see a log of what was sent and received. After the confirmation of receipt of data had been sent, the controller then compared the received data to several case statements, and if the data matched any of those cases then it would complete those cases. If the data did not match any of those cases then the controller would transmit to the PC saying that it received incorrect or invalid data. In the case that the controller received a valid command then the controller toggled the voltage on the I/O pin connected to the signal input on the relay board used to drive the power switching relay. Once the commands in the case statement were completed the microcontroller would then send confirmation to the PC that a correct data sample was received, which actions were taken and the state (on/off) of those switches.

## Operation

Overall the PC can set new voice commands, send switch commands and check inputted voice data against stored commands. This was achieved by a simple text based menu system whenever the command or the corresponding number was written by the user as an input. In Figure 18 the command 7 was sent, which was the quit command.



User input

Commands

Figure 18 – PC command input

The function of the node was to monitor if a command had been sent from the PC over the Bluetooth module. If a command had been received, it was read and compared to a list of actions to take depending on the contents of the command. The main option was to switch a device on/off. When an expected command is received and the action was completed, the node reported this to the PC via the Bluetooth dongle. If data the node did not understand was received, it was reported to the PC.

# Experimental Results

## Audio Capture

There are two audio capture functions written for this program. The first was to capture a voice sample, play it back to the user and then on confirmation save it to a set directory to be used later as a voice command file. The second function constantly sampled the microphone for groups of data of the length of a command. This data was then check against the command files and if there was a match the output of the corresponding pin on the microcontroller was brought high or low to toggle the state of the relay it was connected to. There were problems when implementing these functions. The first was that when the software was programed to start recoding from the Bluetooth headset, the first 4.3 seconds of the recoding would be messed up junk data. The length of this bad data was always 4.3 seconds. To get around this when the command input function was run there was a count down and the user was requested to speak on ‘5’. In truth the recording process had already started but this ensured that good voice data was collected. The first 4.3 seconds of recording was then discarded and the command was played back to the user and the user was asked for confirmation before naming and saving the file.

The second problem was with the function that checks the recorded voice samples against the stored commands. Between the first 4.3 seconds of each recoding being bad data and the time it took for the speech recognition to process, not all speech was being checked. Only one lot of clean data was taken after about every 6 seconds. This made it difficult to speak at the correct time during recoding to get a voice match. To solve this a dynamic time warping function written in C for MATLAB and licensed under a general public license was used [[27](#_ENREF_27)]. Because MATLAB is an interpretive language it was easier to program and more lenient with things such as declarations of variable types. But the down side to that was that it takes allot longer to implement some types of operations. This was not true for the C language, and the C function used increased the speed of the DTW operation by almost a 100 times.

This did not remove the time lag caused by the first 4.3 seconds of every sample being bad data. To remove this rather than taking voice samples over and over for a user inputted number of loops, the code was changed to constantly record for a set amount of time. This meant that if the amount of time for each recording was set to a very long amount of time, there would only be junk data right at the very start of that time and good data for the rest. The MATLAB record function contained a property that allowed for a set action to occur at a set interval of recording. This interval was set to the length of a command file. Then every time that amount of recording had occurred, without stopping recording, the data for that period was checked against the command data using the new and faster dynamic time warping function. This still caused some problems because of memory usage and reading from something that was also being recorded to but was a better solution then the 4.3 second pause before any good data could be received.

If there was access to MATLAB 2012 then a Bluetooth device object could have been created for the Bluetooth headset rather than treating it as an audio device. This would have allowed for button input from the Bluetooth headset to signal when a command was being sent. If this was possible a tone could have been sent to the headset every second for 5 seconds to tell the user when they can say the command. This would remove the need to always constantly check the Bluetooth headsets input and would save on the headsets power and computational time on the PC.

## DTW Tests

These tests were to test the implementation of the dynamic time warping used for speech recognition in this project. Two files were compared for each test result, some were exact matches, some with amplitude, delays, speed or tempo changes (Figures 20 – 33). The purpose of this testing was to make sure the recognition was working, as well as calibrate how much leeway was necessary in recognition for it to be practical and to observe how different known changes in compared files affected the minimum distance plot. The free audio file editor, ‘Audacity’ was used to change the amplitude, phase, tempo and speed of a file set amounts for the purpose of the tests (Figure 19). The difference between using the change tempo and change speed tool in this program was that the change tempo tool attempted to keep the original pitch of the audio file while the change speed tool did not. This causes files to sound similar only sped up when using the change tempo tool and files that sound like a slowed down record player when the change speed tool is used.

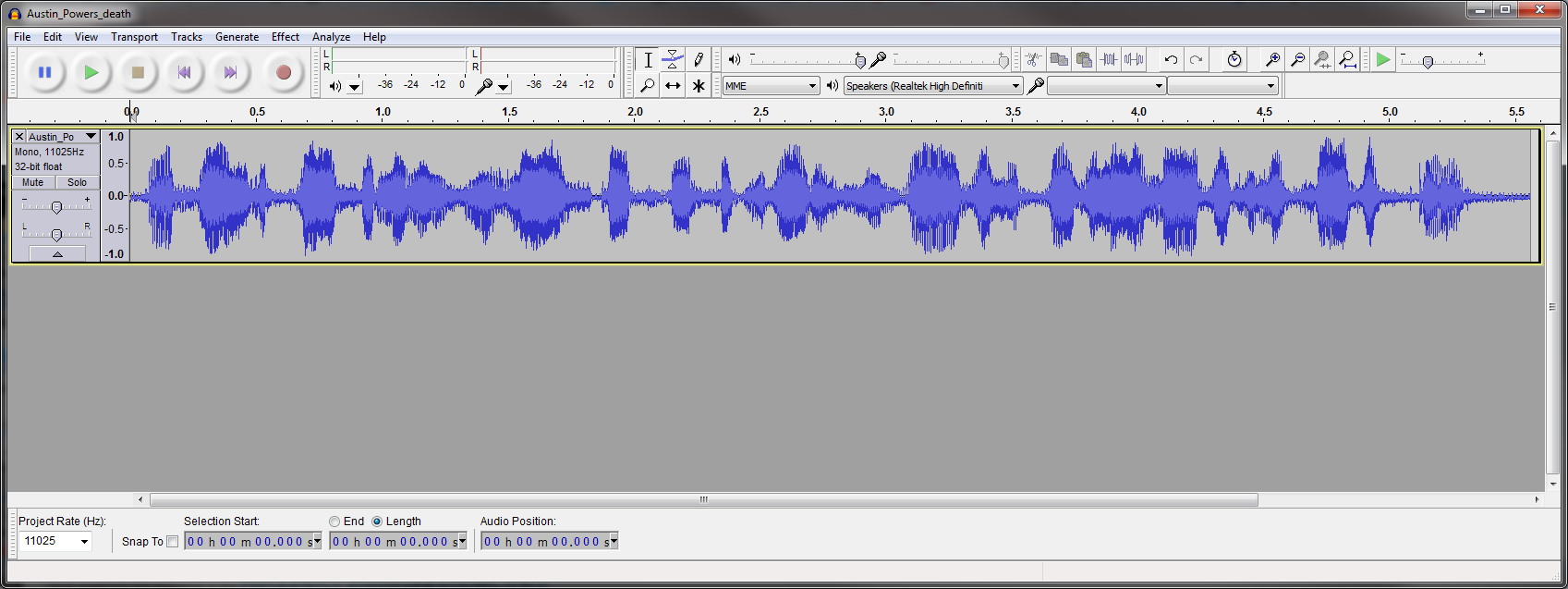


Figure 19 – Audacity being used to edit the ‘Austain Powers voice sample’

### Exact Match

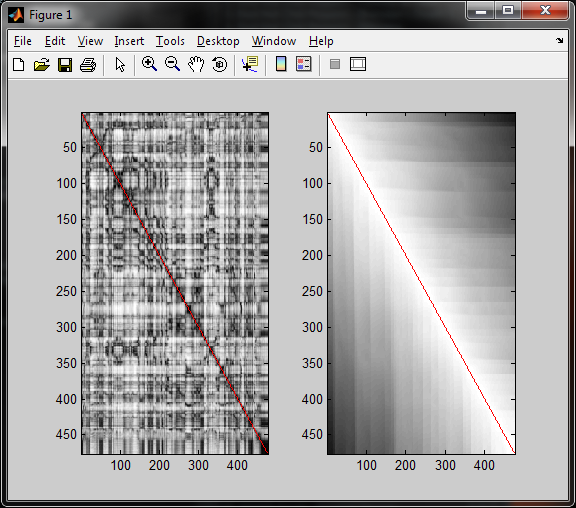


Figure 20 – Two exact matched compared to each other

A single file was read twice and the arrays that resulted were compared to each other. Both arrays were identical. This was to examine the results of the program and difference distance plot when it was at its smallest possible value (an exact match).

### No Match

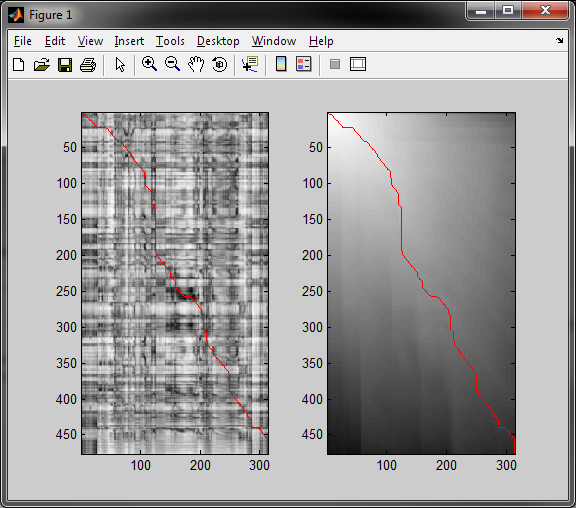


Figure 21 – Two different audio samples compared to each other

In Figure 21 the minimum distance difference path was not strait. This was because the two files compared for recognition were two completely different files.

### Delay

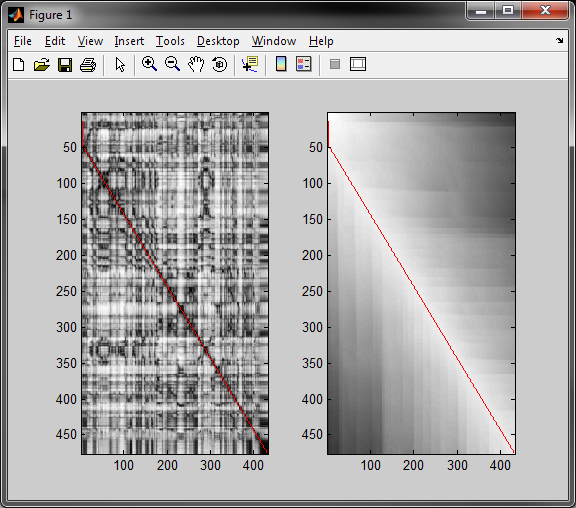


Figure 22 – Delay of -0.5s from original

The effect of a delay was the vertical distance seen in red at the beginning of the minimum distance path in Figures 22 and 23. This distance corresponds linearly with the amount of delay between the original file and the comparison file. The rest of the minimum distance path travelled in almost a strait path to the opposite corner of the plot. This indicated that if detection of a vertical shift at the start of the plot could be formulated then delays at the start of voice data would not affect recognition between two samples.

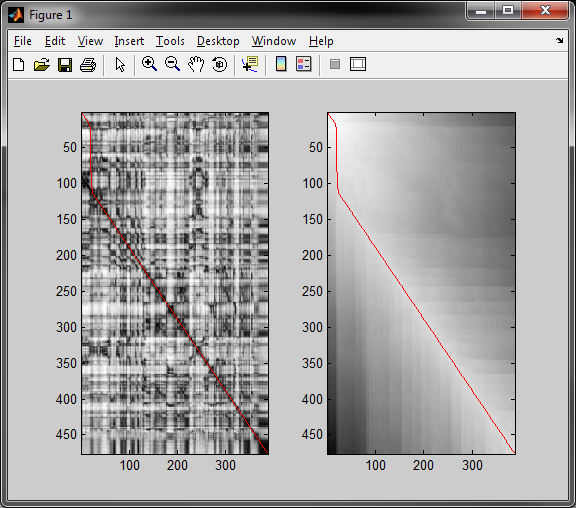


Figure 23 – Delay of -1s from original

### Amplitude Change

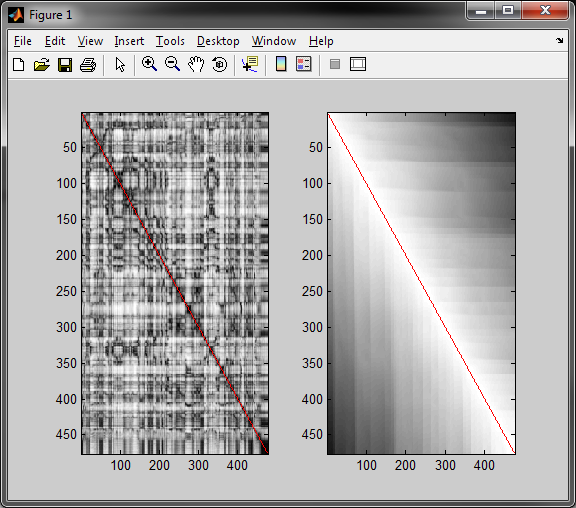


Figure 24 – Decrease of 10dB attenuation

Changes in the level of sound were made to the test file and compared to the original. This was to test if voice data inputted more quietly would have an effect on the recognition process. As long as the measure of ‘loudness’ the command is spoken isn’t so low the microphone was missing, it had no noticeable effect on the speech recognition (Figures 24 and 25).

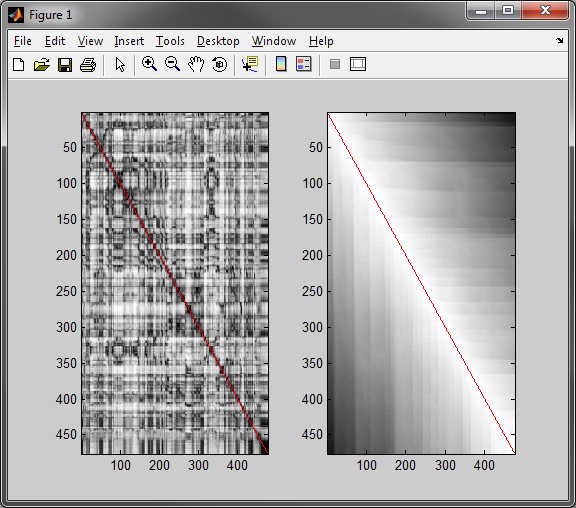


Figure 25 – Decrease of 20dB attenuation

### Tempo Decrease

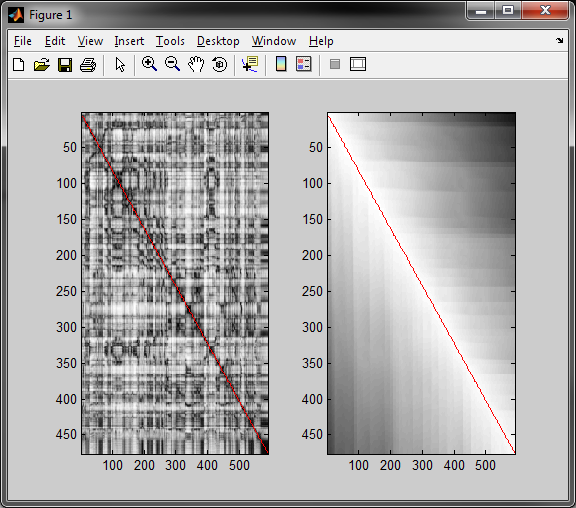


Figure 26 – Decreased tempo by 20 percent

The tempo of a copy of the original file was decreased and increased by 20 and 40 percent and then each was tested for recognition against the original file (Figures 26 – 29). The minimum difference distance plot in all four figures showed a line traveling almost directly to the opposite corner of the plot. This indicated a very close match from the recognition.

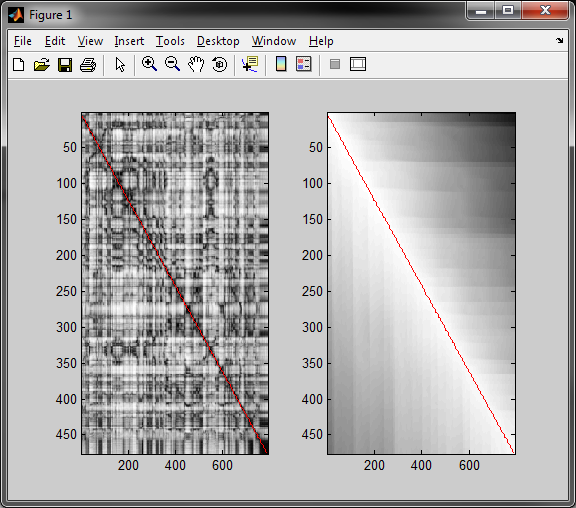


Figure 27 – Decreased tempo by 40 percent

### Tempo Increase

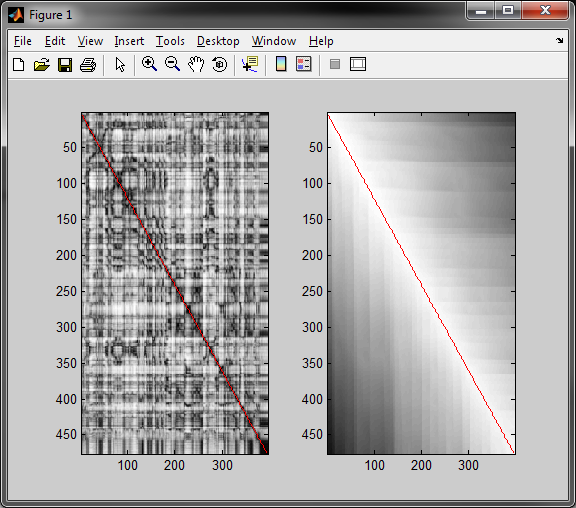


Figure 28 – Increased tempo by 20 percent

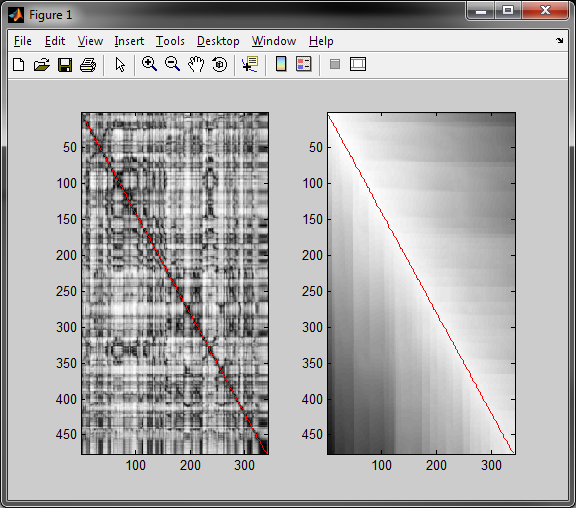


Figure 29 – Increased tempo by 40 percent

### Speed decrease

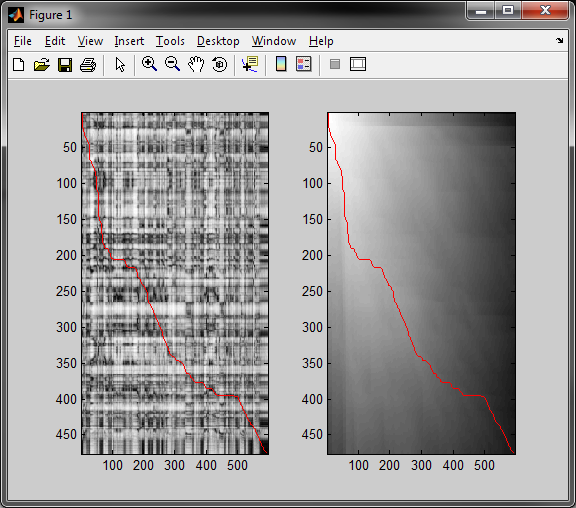


Figure 30 – Decreased speed by 20 percent

The speed of a copy of the original file was then decreased and increased by 20 and 40 percent and then each was tested for recognition against the original file (Figures 30 – 33) similarly to how changes in tempo where examined. The minimum difference distance plot in all four figures showed a line traveling that did not travel directly to the opposite corner of the plot. This indicated changes in speed where the pitch of the original file was not maintained had poor recognition results.

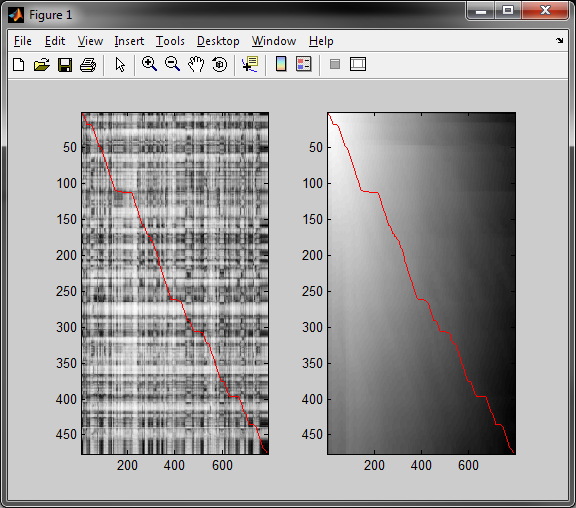


Figure 31 – Decreased speed by 40 percent

### Speed Decrease

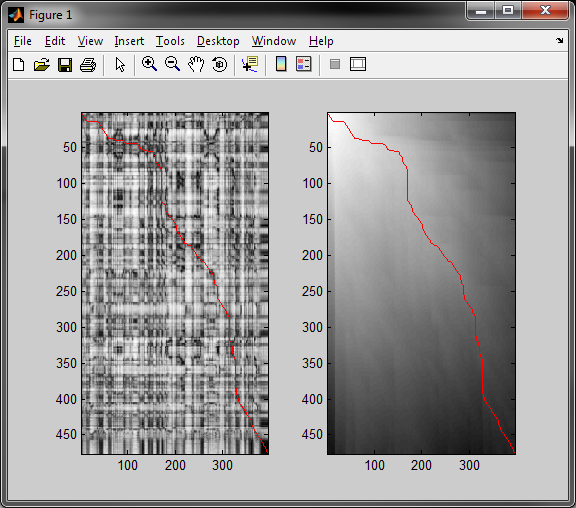


Figure 32 – Increased speed by 20 percent

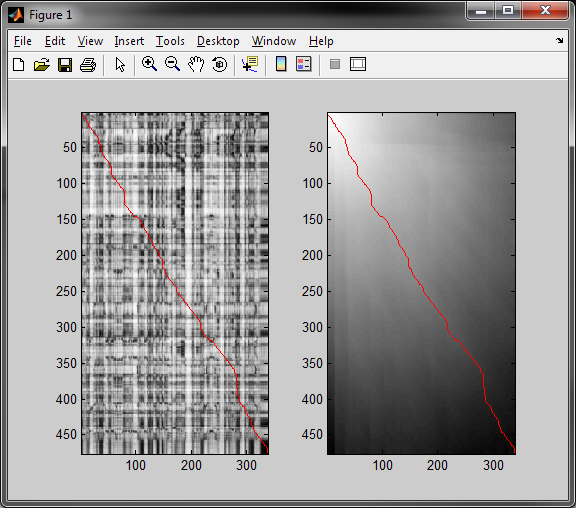


Figure 33 – Increased speed by 40 percent

### DTW Results

From these results it could be seen that changes in the amplitude of a command did not have an effect on recognition when compared to the original. The comparisons that contained the delay could be seen to have a time delay before the data starts to match the original, but from that point on the match was very accurate. Changes in the tempo of voice data resulted in only a very small amount of difference to the original file whereas changes in speed alone caused a significant miss match. This is partly due to the slowed down files having the ends of the file cut off from it being longer than the command data it was being checked against. This showed that the volume of the voice data that was inputted was not significant as long as the microphone could pick it up. If the audio capture started part way through someone inputting voice data, and the voice data was a match, this could be accounted for using checks in the programing because there was not much distance difference between the two files. If the phase shift was significantly large though, so would the difference in the match and the data would not be recognised as a command.

## Switching

Commands SwitchCommand1 to SwitchCommand5 when selected from the PC side menu system toggled the outputs of ports 2.1 to 2.5 on the microcontroller. This was tested by attaching LED’s to these to light up and show when a pins output was pulled high. In the case that there was a recognised match between command one and inputted voice data, SwitchCommand1 was sent to the controller toggling port 2.1. The same is true for matches with voice commands 2-5 and toggling ports 2.2 to 2.5 respectively. Due to the limited amount of output pins available only 5 pins were used and only 5 voice commands can be stored and checked against. The relay is connected to port 2.1 and is used to power an additional LED so show that the mechanical switching of the relay is functioning correctly. When inputted voice data matched command data one three and four, LED’s one three and four were turned on, and the relay attached to port 2.1 was toggled as well (Figure 34).

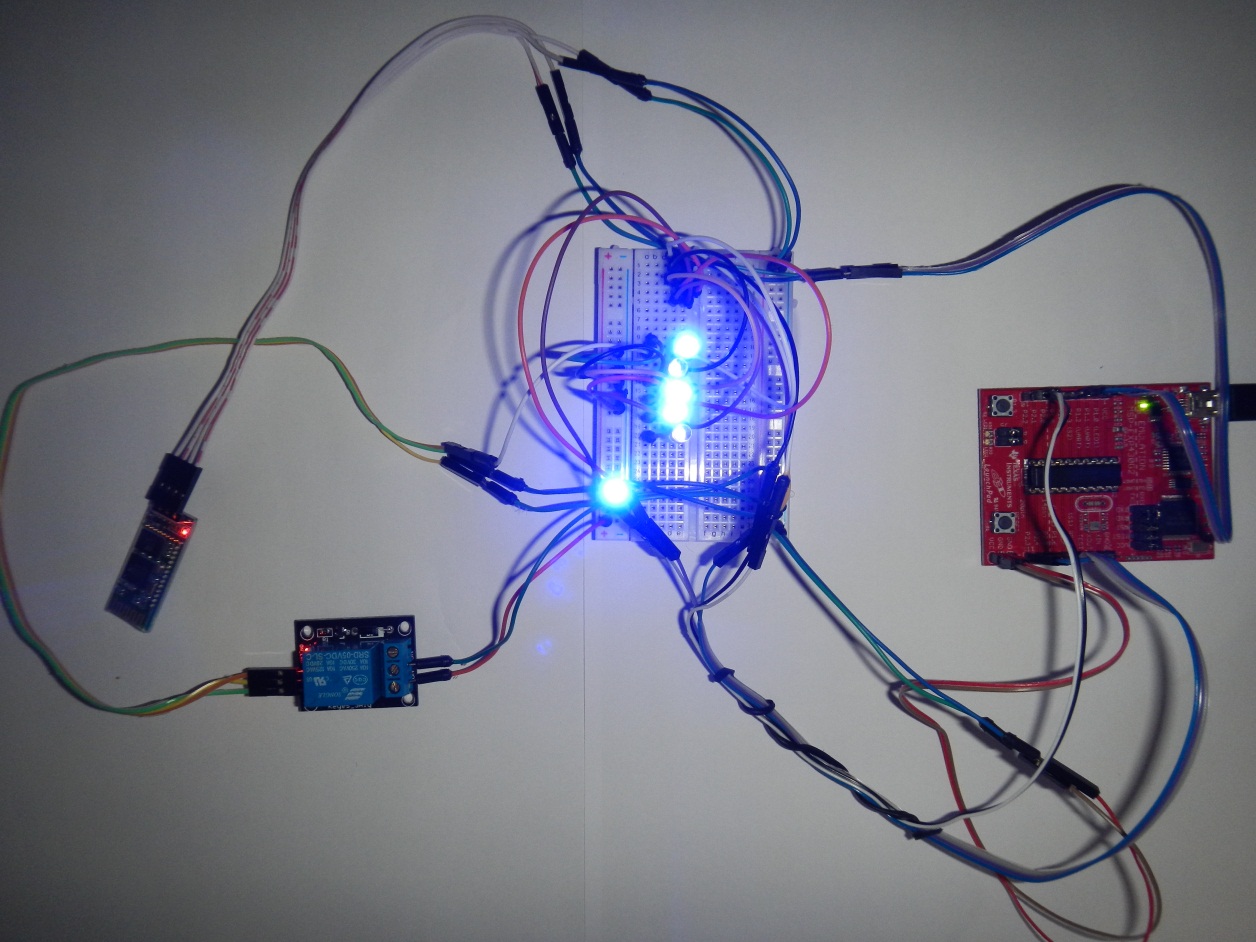


Figure 34 – Node breadboard design, results of a match with voice command 1 and menu options SwitchCommand3 and 4 beings sent to the controller.

# Discussion and Conclusions

## Functionality

The results of this project meet the majority of goals set out to be completed during the proposal stage. The goal of this project was to design a voice controlled automation system for a disabled person’s home. This was a very open ended design constraint, allowing for allot of room for design choices for the project. The only requirements of the project specifications were that the design produced could be controlled using speech recognition and that it automated a process for a disabled person. As there are many kinds of disabilities both mental and physical which require or could benefit from a huge range of different types of support, the goal of the project was to design a solution that could be of use to the largest amount of people possible.

The design decided upon during the proposal stage of this project was to develop an unobtrusive wireless voice controlled power controlling node and a smart phone tie in. As the majority of helpful electronics and automation systems require power to run, this design could allow for any of these systems to be run on voice command. The phone tie in was primarily as a means to be able to trigger emergency calls in cases of need.

The first section of the goals was completed. The result of this project was a node which could communicate with a PC over Bluetooth and control a power supply depending on various commands. The PC side of the project was able to receive and record audio from a Bluetooth microphone, save these files as commands, compare inputted or previously saved voice data to these command or tell the node to switch power without and speech recognition. There were three main short comings for the design, the first was the usability of setting up the device to work properly the first time it was used, and this is covered in the ‘usability’ section of the report. The other two short comings of the design were the range of the Bluetooth devices being limited to approximately 10 meters and that no phone tie in application was completed. Solutions to the Bluetooth range problem are discussed in the section ‘recommendations’. The tie in for phone communication was not completed due to time constraints. There were however applications available for both android and iPhone for controlling the phone via commands sent over Bluetooth [[28](#_ENREF_28)]. There were also serial apps for the android phone that can send commands from Bluetooth to a paired device [[28](#_ENREF_28)], which would allow control of the node from a phone. This would be limited to sending a command and making the node switch a power source.

## Usability

The final design was functional and met the design criteria for the project. Its usability was not as high as it could have been though. During the proposal stage it was soon recognised that some of the people with disabilities who might utilise this design may not have the required knowledge or ability to operate it. For this reason it was recognised that as much error handling as possible should be included in the design, the design should more of less completely set itself up to run and the user interface should be clean and easy to operate. Getting the voice capturing and speech recognition parts of the project working took longer than expected and so the design of the GUI suffered.

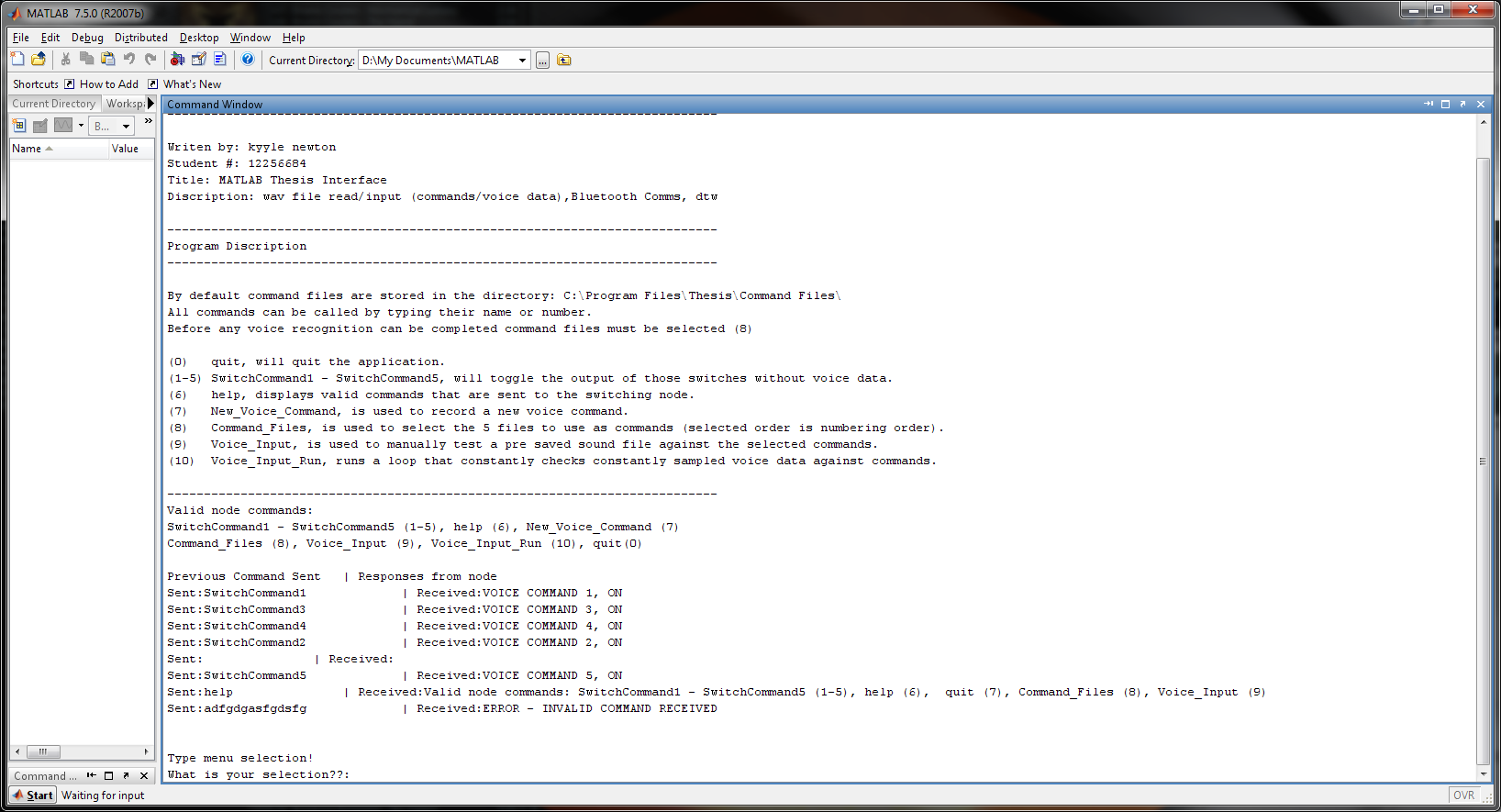


Figure 35 – Final version of the projects GUI

The GUI for the project was still simple to follow and relatively easy to use but required being able to read lines of text and type rather than having large easily visible and accessible buttons for operation (Figure 35). The other major problem with usability was that depending on how the PC the software was running on was set up, comm port allocation and making Bluetooth connections to the PC could be different. Communication with the node was dependent on making a Bluetooth connection and being able to obtain the comm port of that serial connection. Using various checks and coding it is possible to assign a comm port automatically but how the port is set up or if it is connected at all would be dependent on that PC’s Bluetooth software. This means that to initially set up the software someone with computer experience would be required. The .m files that the program and functions are written have been compiled into a package that can be run without MATLAB being installed on the PC.

## Market

This device would have a large amount of marketability if the design was further improved upon and completed. There are already power boards which monitor power usage or include timer circuits which are able to cut the power to devices after a set amount of time. These devices often cost upwards of $70 or as much as $200 to $300 for a full power board [[29](#_ENREF_29)]. By building the Bluetooth switching node and relays into a power board it would enable control and communication from a PC. The computational power of a PC is very large and many different types of functions could be designed by a producer for controlling the devices connected to the power board rather than only the speech recognition implemented in this project. Improving upon mechanical timers, the PC could be set to select which hours in a day particular devices are run or even turned on or off from conditional logic such as at set temperatures. The cost of power is rising and people are becoming more and more familiar with and adoptive of new technologies. The awareness of the costs of high power usage and of the environment is increasing. The cost of all the node components when bought singularly ranges from $15-$20. The idea of futuristic easily implemented home automation that can also save on power costs while improving the quality of life at home is very appealing. If some of the improvements outlined in the ‘recommendations’ section were implemented and the product could be produced at a low enough cost, the design could have a high amount of sales.

### Recommendations

There were several recommendations of improvements that could be made to the design. The first of which would be to improve usability on the PC side of the program. Implement a clean and easy to see and understand user interface. Have the software implement its own Bluetooth connecting code. This was not done in this project as only the 2012 version of MATLAB contains the required functionality without spending very long amounts of time writing unique functions for accessing registry and device information. If the program is able to open Bluetooth connections and the name of the Bluetooth device is known then automatically selecting the correct comms ports is a simple task. These actions would take out the difficulty and requirement of computer knowledge from the installation and setup of this software.

The second recommendation is to incorporate the node and several relays into a ‘smart’ surge protection power board. The microcontrollers used should be one of the MSP430G2553IRSA16R because they are smaller than the model used in development of the project. Solid state relays should be used if they can be sourced cheaply enough because they will not require a constant signal to remain in a switched state, only a pulse to toggle states, which will save on power consumption.

It is recomended to build pre-defined functions for conditions of switching on or off the power that the user can select for each socket of the power board. These could include timer circuits, at specific hours of the day or turning on or off if another device is turned on or off.

A phone app that is used to communicate with the node or PC via Bluetooth would add additional functionality that was going to initially be designed for this project. In essence this would simply be a menu system that then sends commands via serial Bluetooth to the node or PC to interpret and vise versa. This would allow the PC to control the phone with set interpreted commands, or the phone to control the PC. Not only could it be used for emergency calls under set conditions, but smart phones are becoming more and more of a part of our day to day lives. The two major brands behind smart phones are Google and Apple, which both have a huge following.

Similarly the addition of Facebook/Twitter integration would add functionality and appeal. There are millions of users of both Twitter and Facebook. Adding a function that checked a user’s Facebook or twitter for commands for controller the users’ things at home would add an even higher level of usability and control to the automation system as well as appeal to younger generations. The control and commands themselves could be the same as those used for the Bluetooth phone control except that they are communicated on a Facebook app or on Twitter over the internet.

The sixth recommendation is to deal with the limited range of Bluetooth devices. This could be achieved by adding overhead to each transmission that contains the ID of where the transmission needs to go. Each node passes it to the devices that they are connected to. If there ID is there’s it completes the command. If it was not for it the node sends it on. An original time stamp would need to be added to the transmissions overhead so that each node could check if it had already received and sent on that command, to prevent the same command circling around between the nodes or being received twice..

The final recommendation is to use the above implementation to allow for functionality to pass voice data from node to node back to the PC. The voice recognition samples are only a few seconds long each but all sound heard by the microphone is checked by the PC, so transfer rates could be a problem. The newer versions of Bluetooth are design to handle transmissions of that type so it should be possible.

## Conclusion

The aim of this project was to form a design that was reasonably cost effective, modular, adaptive, easy to implement, easy to operate, has a non-invasive installation, and was suitable for operation by people suffering from the widest range of problems. The advantages and disadvantages of each topic covered in the literature review were reviewed with the aim of giving the best range of desired characteristics for the system.

The approach outlined in chapter 4 was designed to solve these problems. As can be seen in chapter 5 not all of these goals were met and not all the approaches in chapter 4 were used. In practice during this projects development there were many unforeseen complications that required additional design and code modifications to mitigate. The outlined overall methodology proposed going into this project was sound in approach, goals and principle but unforeseen problems such as the audio capturing problem caused delays in time and required new methods to find a solution. Some of the goals of this project were not completed in the time available but are still sound goals and features for this type of design. These goals such as the phone integration have been added to the recommendations section of this document for further development at a later time.

The final functional design prototype completed during this project was modular though only a single node was created. The coding on the PC was written using several main functions to make the software design modular and adaptable for future development. The interface while not of the standard format used in most applications was still easy to use and understand. All audio input and node control was completed over Bluetooth communications making the design non-invasive for installation. The project design itself would require someone with a standard level of technical experience to set up the device at its current level of development but could be operated by anyone. As the device is capable of controlling when any electronically powered automation device is turned on or off it is capable of being of use to people in a very large range of situations. In conclusion the final functional design solution of this project met both the goals and design criteria of the project.

# References

[1] D. Finch, "Remote Speech Activated System," Bachelor of Engineering in Computer Systems Engineering Undergraduate Thesis, James Cook University, 2005.

[2] P. Saunders, "The costs of disability and the incidence of poverty," *Australian Journal of Social Issues,* vol. 42, pp. 461-480, 2007.

[3] S. Mitra, P. A. Findley, and U. Sambamoorthi, "Health Care Expenditures of Living With a Disability: Total Expenditures, Out-of-Pocket Expenses, and Burden, 1996 to 2004," *Archives of Physical Medicine and Rehabilitation,* vol. 90, pp. 1532-1540, 2009.

[4] M. LaPorte, "A place to call home. Assisted living providers focus on specialty care," *Provider (Washington, D.C.),* vol. 37, pp. 22-24, 27-2431, 2011.

[5] C. S. Kim, O. Lee, J. Lee, S. Kwon, and M. K. Chung, "Effects of ramp slope and height on usability and physiology during wheelchair driving," San Francisco, CA, 2010, pp. 698-702.

[6] D. C. Balmer, "Impact of the A18.1 ASME standard on platform lifts and stairway chairlifts on accessibility and usability," *Assistive Technology,* vol. 22, pp. 46-50, 2010.

[7] J. D. Enderle, "Annual national science foundation publication on senior design projects to aid persons with disabilities," *American Society of Mechanical Engineers, Bioengineering Division (Publication) BED,* vol. 43, pp. 271-272, 1999.

[8] S. Longhi, C. Nicoletti, and L. Platt, "Interpreting wage gaps of disabled men: The roles of productivity and of discrimination," *Southern Economic Journal,* vol. 78, pp. 931-953, 2012.

[9] J. Cullinan, B. Gannon, and E. O'Shea, "The welfare implications of disability for older people in Ireland," *European Journal of Health Economics,* pp. 1-13, 2011.

[10] D. P. Doessel and R. F. G. Williams, "Disabled people's living standards: Filling a policy vacuum," *International Journal of Social Economics,* vol. 38, pp. 341-357, 2011.

[11] I. P. Donald, "Housing and health care for older people," *Age and Ageing,* vol. 38, pp. 364-367, 2009.

[12] D. J. Wilson, J. M. Mitchell, B. J. Kemp, R. H. Adkins, and W. Mann, "Effects of assistive technology on functional decline in people aging with a disability," *Assistive technology : the official journal of RESNA,* vol. 21, pp. 208-217, 2009.

[13] H. Publishers., "Collins English Dictionary - Complete & Unabridged 10th Edition.," in *Collins English Dictionary - Complete & Unabridged 10th Edition.*, ed. HarperCollins Publishers.: HarperCollins Publishers., 2012.

[14] B. Baker. (2004, April 20th, 2012). *Stephanie - Voice Controlled Home Automation System*. Available: <http://absolutemaximumratings.com/projects_stephanie.php>

[15] A. R. Al-Ali and M. Al-Rousan, "Java-based home automation system," *IEEE Transactions on Consumer Electronics,* vol. 50, pp. 498-504, 2004.

[16] I. Coşkun and H. Ardam, "A remote controller for home and office appliances by telephone," *IEEE Transactions on Consumer Electronics,* vol. 44, pp. 1291-1297, 1998.

[17] H. V. Dange and V. K. Gondi, "Powerline communication based home automation and electricity distribution system," Coimbatore, 2011.

[18] Y. Tang, L. Yue, and S. Zhu, "Communication protocol of home network based on power line carrier technology," Wuhan, 2011.

[19] A. Gupta, A. Mudgal, C. Jayaraj, J. Kulkarni, K. Jain, P. Murarka, and S. Gujar, "Smart home device and energy management systems," Hyderabad, 2011.

[20] R. Piyare and M. Tazil, "Bluetooth based home automation system using cell phone," Singapore, 2011, pp. 192-195.

[21] K. Gill, S. H. Yang, F. Yao, and X. Lu, "A ZigBee-based home automation system," *IEEE Transactions on Consumer Electronics,* vol. 55, pp. 422-430, 2009.

[22] C. Luo, Y. Chen, M. Krishnan, and M. Paulik, "The Magic Glove: A gesture-based remote controller for intelligent mobile robots," Burlingame, CA, 2012.

[23] C. Bayilmiş and E. Kelebekler, "Remote control of a CAN-based mobile model car using a voice activated control system," in *CAN temelli gezgin bir model aracin ses aktivasyon sistemi ile kablosuz kontrolü*, Aydin, 2008.

[24] G. C. S. Ruppert, P. H. J. Amorim, T. F. Moraes, and J. V. L. Silva, "Touchless gesture user interface for 3D visualization using the Kinect platform and open-source frameworks," Leiria, 2012, pp. 215-219.

[25] H. A. Caltenco, B. Breidegard, B. Jönsson, and L. N. S. Andreasen Struijk, "Understanding Computer Users With Tetraplegia: Survey of Assistive Technology Users," *International Journal of Human-Computer Interaction,* vol. 28, pp. 258-268, 2012.

[26] MathWorks. (2012, 24/9/2012). *wavread* Available: <http://www.mathworks.com.au/help/matlab/ref/wavread.html;jsessionid=9d19af59ec692ac0bc90a28138ad>

[27] D. Ellis. (2006, 19/9/2012). *Matlab Audio Processing Examples* Available: <http://labrosa.ee.columbia.edu/matlab/>

[28] P. D. T. P. S.L. (2010). *BlueTerm*. Available: https://play.google.com/store/apps/details?id=es.pymasde.blueterm&hl=en

[29] C. Simms. (May 14, 2012). *Save power with smart-sensing power boards and timers*. Available: <http://www.cnet.com.au/save-power-with-smart-sensing-power-boards-and-timers-339337787.htm>