Department of Electrical and Computer Engineering

Wireless Infrared Headset Communications for Vision Impaired Persons in Multi-User Environments

Andrew Pasquale

This thesis is presented for the Degree of
Master of Engineering
of
Curtin University of Technology

November 2009

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

Signature: ………………………………………….

Date: ………………………...

Abstract

A dual communications systems with application to call centre environments has been developed to aid visually impaired persons. The needs of the visually impaired in such a situation are for wire-less communications as wires connecting the headphone easily become tangled and this is a difficulty for the visually impaired to rectify and is distracting from their work. The proposed system utilises infrared communications for much of the normal traffic. It is stereo with one channel devoted to the enquirer audio channel and one to an aural description of the computer screen being used to perform the call centre actions. This system needs to be high bandwidth, but to achieve that in this environment demands directivity. Hence if the the operator turns away from the screen they lose connectivity. To overcome that, a second, lower bandwidth radio frequency communications system is proposed that keeps the operator in touch with call centre operations. A working test bed was created to assess the suitability of the infrared communications in such a headset. It was found found that it performed satisfactorily within the environment although its effectivness was limited by the users' natural body movements. This was more than compensated for by its superior security and sound quality.

Acknowledgements

I would like to thank Dr Douglas Myers (supervisor) and Mr Iain Murray (co-supervisor) for their continued help, patience, support and guidance in the course of this research. This research would not exist without their initiative and much of the work completed must be credited to their direction. Thanks to John Heppel, Mark Fowler, Serge Mokrous and Daniel Jackson for their practical guidance and use of tools without want of thanks. I would also like to thank my wife, Brenella Pasquale, for her continued support, encouragement and patience during the course of this project.

Table Of Contents

List of Figures

List of tables

List of acronyms and abbreviations

ACELP:	Algebraic Code Excited Linear Prediction
ACK:	Acknowledgement (packet/frame used in networking)
ACL:	Asynchronous Connection-Less
ACS:	ACoustic shock Syndrome
ADT:	Angle Diversity Transceivers
ASK:	Amplitude Shift Keying
ATC:	Adaptive Threshold Control
AP:	Access Point
BER:	Bit-Error Rate
BT:	Bluetooth™
CDMA:	Code Division Multiple Access
CELP:	Code Excited Linear Prediction
CIL:	Confidence Interval Lower
CIU:	Confidence Interval Upper
COTS:	Commercial Off The Shelf
CRC:	Cyclic Redundancy Check
CRT:	Cathode Ray Tube
CVSD:	Continuously Variable Slope Delta
DAA:	Data Access Arrangement (an interface circuit to the PSTN)
DAC:	Digital to Analog Converter
DECT:	Digital Enhanced Cordless Telecommunications
DSP:	Digital Signal Processor
DSSS:	Direct-Sequence Spread Spectrum

EMI:	Electro-Magnetic Interference
FDMA:	Frequency Division Multiple Access
FER:	Frame Erasure Rate
FHSS:	Frequency Hopping Spread Spectrum
Flash:	A form of non-volatile computer memory
FoV:	Field of View
GMSK:	Gaussian Mask Shift Keying
IDE:	Integrated Development Environment
IEEE:	Institute of Electrical and Electronic Engineers
IR:	InfraRed
IrDA:	Infrared Data Association
ISM:	Industrial, Scientific and Medical
ISO:	International Standards Organisation
JTAG:	Joint Task Action Group (a standard programming port interface for the DSPs and microcontrollers)
LCD:	Liquid Crystal Display
LoS:	Line of Sight
LSB:	Least Significan Byte
MAC:	Medium Access Control layer
MIMO:	Multiple In Multiple Out
MOS:	Mean Opinion Score
MSB:	Most Significan Byte
MSD:	Multi-Spot Diffusing
NFMC:	Near Field Magnetic Communication

OFDMA:	Orthogonal Frequency Division Multiple Access
OSI:	International Organisation for Standardisation
OTP:	One Time Pad
PAN:	Personal Area Network
PCB:	Printed Circuit Board
PCM:	Pulse Code Modulation
PER:	Packet Error Rate
PESQ:	Perceptual Evaluation of Speech-Quality
PHY:	Physical layer
PPM:	Pulse Position Modulation
PSK:	Phase Shift Keying
PSTN:	Public Switched Telephone Network
QAM:	Quadrature Amplitude Modulation
QoS:	Quality of Service
RBER:	Residual Bit Error Rates
RF:	Radio Frequency
RFID:	Radio Frequency Identification Device
RSSI:	Received Signal Strength Indicator
RTS:	Request To Send
RTT:	Round-Trip Time
Rx:	Receive / Reciever
RZI:	Return to Zero Inverted
SCO	Synchronous Connection-Oriented
SDT:	Spatially Diverse Transceivers

SNR:	Signal-to-Noise Ratio
SOT:	Self Orientating Transcievers
TCM:	Trellis Coded Modulation
TDMA:	Time Division Multiple Access
Tx:	Transmit / Transmitter
UWB:	Ultra Wide Band
VAD:	Voice Activity Detection
VIP:	Vision Impaired Person
VOIP:	Voice Over Internet Protocol
WDMA:	Wavelength Division Multiple Access
WLAN:	Wireless Local Area Network

1.0 Introduction

1.1 Enabling the vision impaired in the workplace.

There are about 16 million persons of working age (15-64 years) in Australia, of whom approximately 109,000 are blind or vision impaired. These vision impaired persons (VIPs) have great difficulty in obtaining meaningful employment. Figures from 2007 showed about 69.2% were unemployed, as compared to 35.4% for the national average (ABS 2007). In a bid to gain employment, people with vision disabilities often seek positions in call centres where their disability is not a significant disadvantage. However, difficulties with the cable and cable jacks used with audio headsets results in the employment opportunities being not as great as they might be (Hill, 2002). A solution to this problem is wireless headsets. Presently available solutions though are intended for single user or low density environments, not a high density multi-user environment such as a large call centre.

Multi-user environments present some challenging problems. In particular, the high number of closely grouped users causes difficulty in providing reliable wireless communications with sufficient audio fidelity. Further, it needs to be dual channel so that the VIP can interact with both the caller and their computer terminal via screen reading software. Privacy demands require that any conversation remain confidential with respect to those within the call centre and any external eavesdroppers.

The use of such wireless headsets is not confined to the vision-impaired. Any multi-user environment dependent on good voice communication can benefit such as occur in defence, conference organisation and sporting events.

1.2 The value of infrared communications for wireless headsets in a multi-user environment

Current wireless headset solutions, the majority being radio based, do not meet the requirements of VIPs working in call centres for reasons including sound quality, security, reliability and scalability. To be suitable, the wireless headset must have high fidelity dual channel audio, secure communications and robust operation given the possible large number of operators.

The headset could use infrared (IR) as a medium for wireless communications instead of radio. This has line-of-sight (LoS) and short propagation distance characteristics, ideal for creating a cellular communication network where users do not interfere with each other, even in a high density environment such as call centres. The use of a cellular network allows high fidelity and reliable wireless communications, and the LoS characteristic will allow security to be easily implemented. However, in practice, the LoS restriction would limit a user's natural human rotational and lateral movements and consequently would be highly unergonomic and uncomfortable to use .

To overcome the limitations of infrared and radio media on their own, a conceptual solution to the call centre problem is a wireless headset combining the infrared medium for base-load communications and a radio frequency (RF) personal area network (PAN). Such a hybrid combination can achieve the following:

- *Audio fidelity and reliablity*. When the user is facing and interacting with their computer terminal, the infrared medium is used in a cellular manner to provide high bandwidth and high fidelity audio. Also, given the cellular nature of the IR network, the communication is immune to interference and is therefore reliable and robust.

- *Ergonomics and ease of use*. When the user rotates or moves around their desk space, the radio PAN is used to pick up and maintain communications which may have been lost by the LoS limited IR link. During these times, when the user is not facing their terminal, it is assumed that the user is unlikely to be interacting with the computer terminal and so will therefore not be requiring robust communications with high bandwidth and high quality audio.
- *Security*. Infrared LoS communication can be made very secure by transmitting audio data along with encryption keys and these may be buffered. When the radio PAN is required to maintain the link, the buffered encryption keys can be used to provide the same or similar level of effective security as one-time-pad (OTP) crytpographic algorithms.

While the IR/RF hybrid combination is potentially an ideal solution, the following issues need to be resolved:

- The effectiveness and ergonomics of LoS (LoS) IR in providing base-load wireless communications
- The practical application and robustness of cellular LoS IR communications in call centre environments

These issues were investigated and a demonstration system created that shows a wireless headset utilising IR for base-load communications may provide a suitable system design for VIPs in multi-user environments. Hence a means to resolve their employment difficulties and overcome the drawbacks they experience with wired headsets.

1.3 An overview of the thesis

Chapter 2 lists the requirements for a wireless headset for VIPs, and in particular for VIPs working in call centres. The chapter also reviews research into the problem.

Chapter 3 reviews commonly available wireless communication media and their suitability to the needs of VIPs working in call centres. The IR/RF combination is further refined as a solution to meet the requirements of the headset.

Chapter 4 introduces the issues surrounding the IR/RF solution and outlines specific elements of the design that needed to be tested and proven. The wireless headset testbed that was developed to investigate the issues raised is then detailed followed by results of the tests undertaken and derived conclusions.

Chapter 5 provides a summary of the findings during the project and a comparison of the IR/RF hybrid design to alternative techniques. Recommendations for future development of the wireless headset design are also discussed.

The appendices include the source code, schematics, pictures of the developed headset and other background information.

2.0 Wireless headset requirements and current solutions

2.1 The need for a wireless headset for VIPs

VIPs cannot interact with the visual information displayed on CRT/LCD monitors in the same way sighted persons are able to do. Therefore, tactile displays and screen reading text-to-speech methods are currently the most practical way of accessing this information.

Tactile displays, such as refreshable Braille displays are a fast and convenient method of information access for VIPs. However, they are greatly limited in their resolution and functionality. These displays are also expensive to purchase and maintain. Figure 2.1 shows an example of these devices.

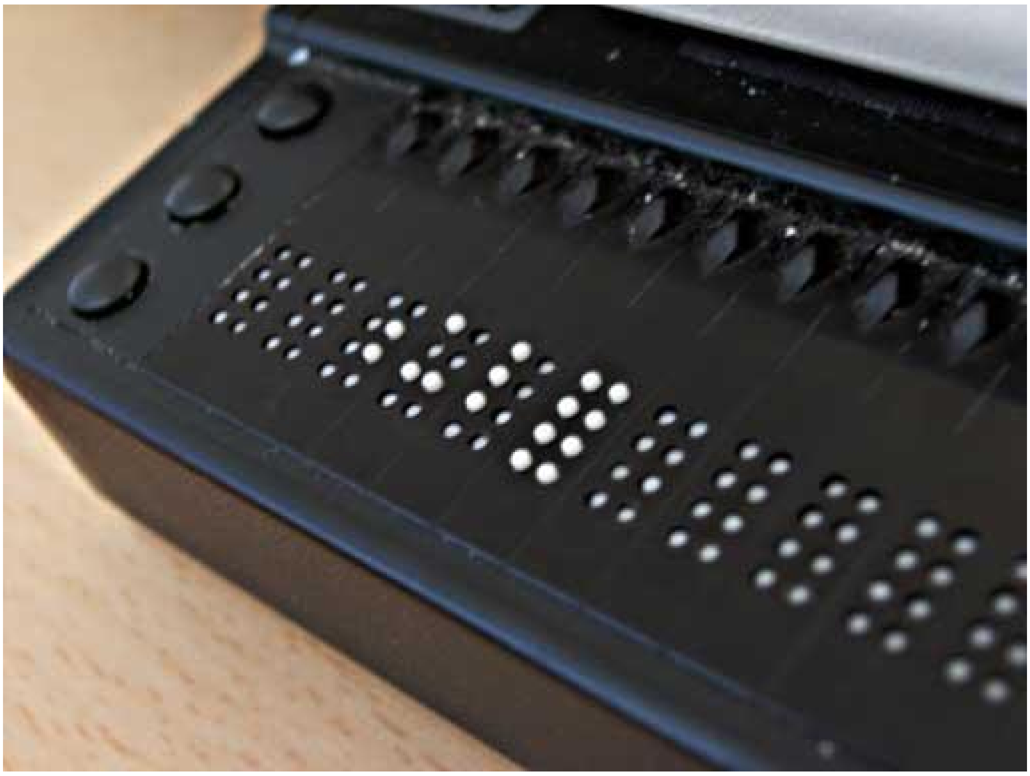


Figure 2-1: Haptic Braille Display (Wikipedia 2008b)

Screen reading software combined with a text-to-speech converter is the most common method of computer information access by VIPs. Commercial products like JAWS (Freedom Scientific 2009) and Apple VoiceOver (Apple 2009) use the accessibility features of the operating system to determine the contents of the current graphical user interface (GUI) and user controlling actions and narrate this to the VIP. Given the large amount of information present in a two dimensional (2D) GUI and a need to work at reasonable pace, VIPs use the screen reading software with very high speech rates, generally 100 words per minute or higher. Hence screen reading software requires high quality and low delay audio for efficient computer interaction.

Screen reading software is typically employed with a dual channel headset with a microphone. This allows the VIP to also communicate with a telephony customer concurrently if they so desire. The problem this presents, though, is that VIPs experience frequent problems, including;

- breakage of headset cables;
- pulling out of audio jacks by Guide dogs;
- tangling of headset wires.

Full details of these problems are outlined in Appendix V.

2.2 Detailed requirements of a headset for use by VIPs

2.2.1 Essential criteria that must be considered

To be suitable for VIPs working in call centre environments a headset must have stereophonic high quality audio, secure communications and such ergonomic features as low weight, small size and long battery life. The following sections describe each of these requirements in detail.

2.2.2 Stereophonic sound

The requirement for a headset with stereophonic sound arises from the call centre environment where the VIP must concurrently listen to the telephony customer and the screen reading software of the computer information source. As described by Murray (2006):

> *"The dual channel audio requirement relates to the scenario that in a call centre VIP's receive voice output from screen reading software on one channel, and a telephone output for human interaction on the other. For user intelligibility the channels must be totally independent and devoid of inter-channel interference. A third channel from the headset to the base station must be incorporated for two way communication with the client caller."*

It is interesting to note that while not scientifically proven, there is a perception that VIPs can develop an above average ability to interact with simultaneous human and computer audio channels compared to non-VIPs.

2.2.3 High quality audio

2.2.3.1 Qualitative requirements

In the call centre scenario, the VIPs interaction with the telephony customer are typically via the Public Switched Telephony Network (PSTN) or an equivalent voice over internet protocol (VOIP) based system. The PSTN and VOIP deployments are typically designed to provide “toll quality” audio that, in Australia, is equivalent to an 8-bit companded signal sampled at 8KHz. The wireless communication to the headset needs to maintain this quality and likewise must return the VIPs voice back into the PSTN or VOIP system at a similar quality.

The requirement for high quality audio stems from the use of screen reading software by VIP’s for computer interaction. The screen reading software verbally communicates the computer information at an extremely high word rate, and if high quality audio communications are not used, the user can become easily ‘hearing fatigued’, a significant problem for an employee that may work an eight hour shift. The major factors affecting speech intelligibility include frequency response, noise (quantization, transmission induced and distortion) and transmission delay.

2.2.3.2 Frequency response

There has been much research conducted on establishing the relationship between spectral response and speech intelligibility. For example, Rodman (2006) with reference to Figure 2-2 stated:

> *“the burst of high-frequency sound that distinguishes the "s" in "sailing" from the "f" in "failing" occurs between 4 kHz and 14 kHz. When these frequencies are removed, no cue remains as to what has been said.”*

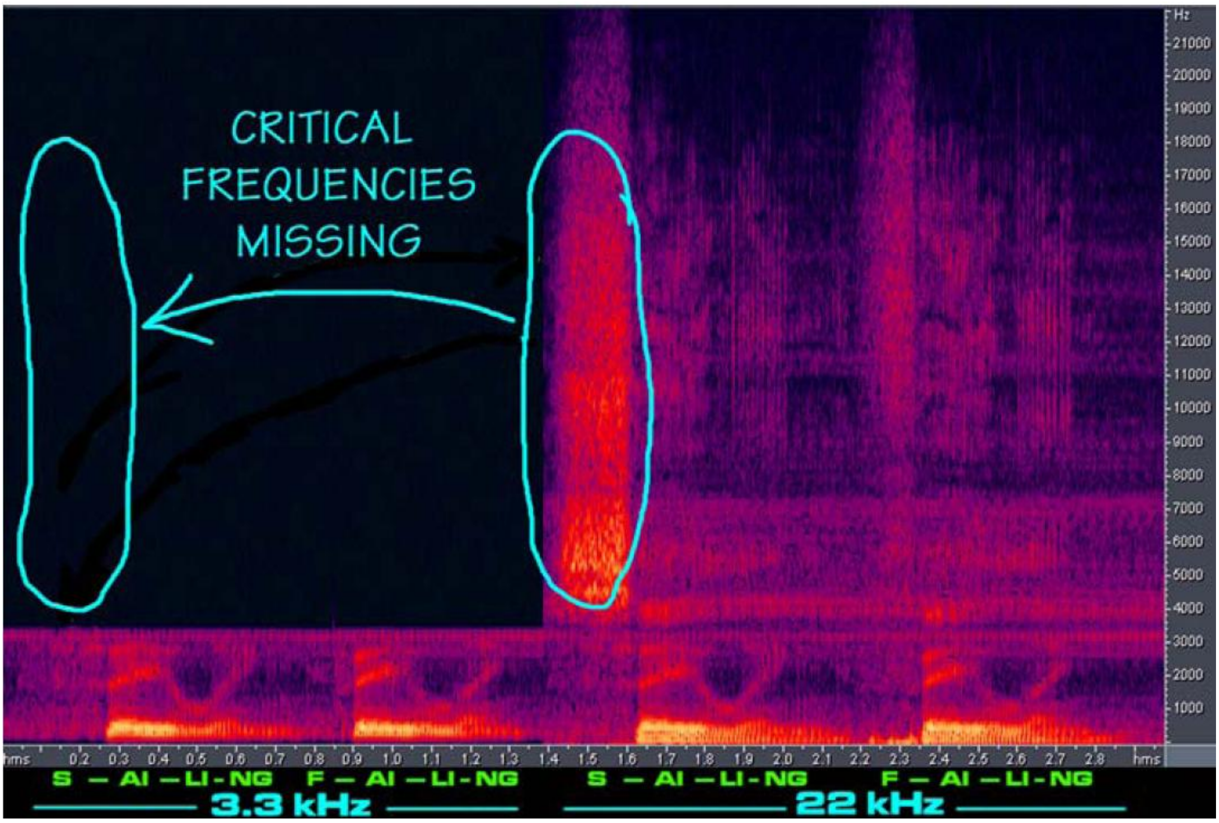


Figure 2-2: Speech Spectra of 'sailing' and failing' at 3.3khz and 22khz (Rodman 2006)

He also highlights:

> *"In normal speech, words come at a rate of about 120 words per minute. Consequently, 3.3 kHz speech produces about 40 ambiguities per minute, where 7 kHz speech will produce fewer than four".*

Certain compression techniques, however, do promise to deliver 'wideband' speech (50-7000Hz) at low bit rates including 23.5kbps Algebraic Code Excited Linear Prediction (ACELP). However the use of compression methods such as ACELP still affects speech intelligibility. As seen in Figure 2-3, mean opinion score (MOS) drops from 4.4 to 3.9 for wideband uncompressed and 23.5kbps compressed speech respectively.

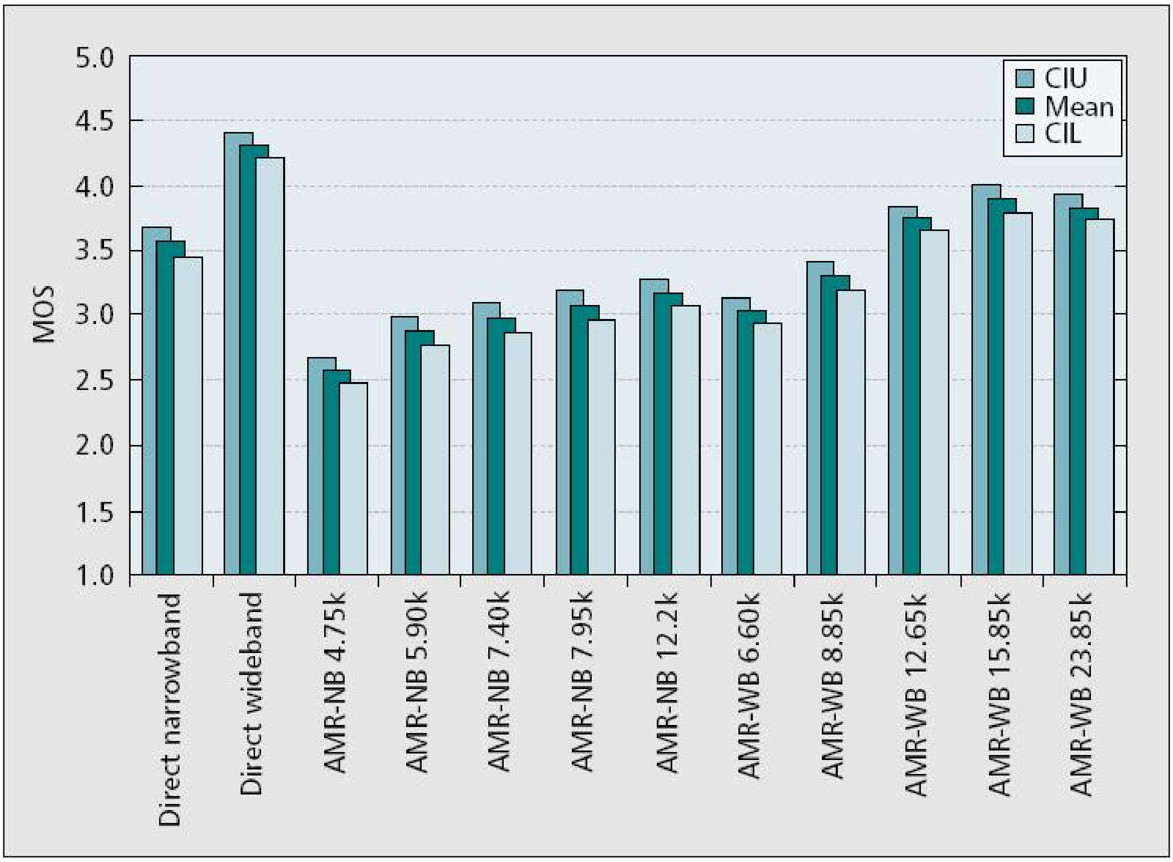


Figure 2-3: Subjective test results comparing AMR narrowband and AMR-WB. (Ojala et al, 2006)

(The CIL and CIU abbreviations stand for 95 percent confidence interval lower and upper boundaries, respectively)

This reduction in MOS for speech intelligibility and the resultant hearing fatigue must be considered for VIPs using the headset for a working day.

To determine the required spectral response of text to speech audio, where this is computer synthesised rather than natural human, samples at typical ‘words per minute’ rates were captured and analysed.

At a relatively slow 60 words per minute, the frequency response of the speech generated by JAWS software with the default vocal tone shows a requirement for a

7KHz frequency response (Figure 2-4), that according to Nyquist’s theorem equates to a 14KHz sampling rate.

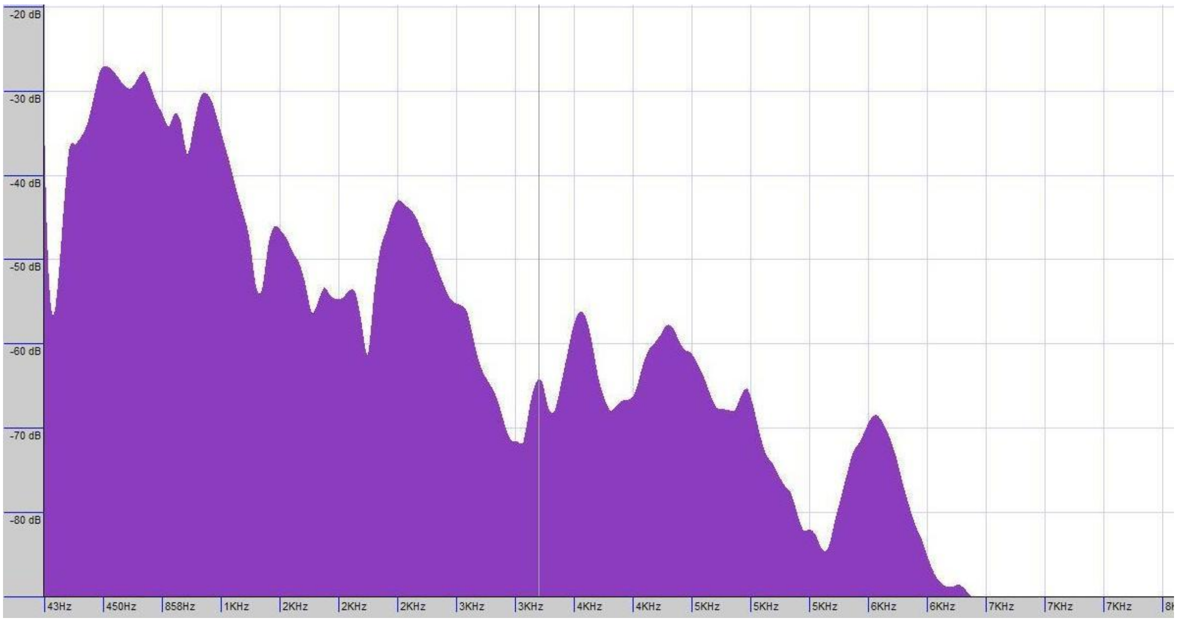


Figure 2-4: Spectral Analysis of Jaws text to speech at 60 words per minute.

At a relatively normal 110 words per minute for the same speech, the frequency response generated by the JAWS software shows additional information present in the 6-8KHz band (Figure 2-5), and hence a requirement for a sampling rate of 16KHz as the standard to be adopted.

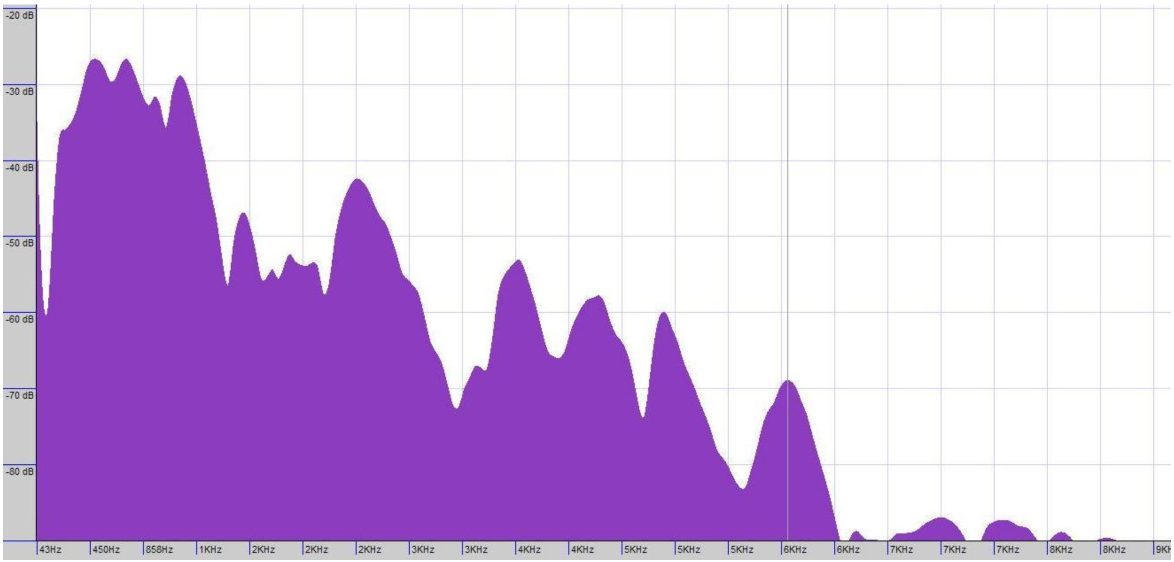


Figure 2-5: Spectral Analysis of Jaws text to speech at 110 words per minute

Performing subjective tests with VIPs revealed a noticeable difference in audio quality between sample rates at 8KHz and 16KHz. While intelligible, VIPs characterised 8KHz operation to have a duller sound than 16KHz operation and required more concentration in use. This confirms the spectral analysis findings (depicted in Figure 2-5) and also that an 8KHz sampling rate results in the loss of important information present above 4KHz.

2.2.3.3 Distortion, quantisation, and transmission noise

The acquisition, digitisation and transmission of audio data can add quantisation noise to the signal and reduce the listener’s ability to decode and interpret the acoustic information. The signal to noise ratio of the audio data presented to the listener, amongst other factors affecting quality, must be considered for a successful headset design.

The electronic hardware used to digitise and transmit the audio must also take care not to distort the information by clipping waveform peaks, introducing frequency dependent phase adjustments or by adding reverberation or echo to the signal. There are various techniques for avoiding these issues and they must be considered in the final design to maintain a high quality audio link.

When an analogue signal is sampled to digital form (digitisation), quantisation noise is introduced. Simply put, there is an inverse relationship between the quantisation noise and the number of bits (information) used to represent the data. As this ultimately forces a compromise between audio fidelity and data bandwidth, careful consideration should be taken during design.

Many studies have been conducted on the effect of quantisation noise on speech intelligibility. However, in order to gain a subjective benchmark of the allowable quantisation noise in high rate speech to text audio, tests were performed with audio presented to VIPs in 8-bit and 16-bit pulse code modulated (PCM) digitisation forms. The tests revealed the expected quantisation noise at 8-bits resolution, especially for quieter sounds. 16-bit audio showed negligible noise and is the desirable standard that will be worked towards. Straight PCM digital representation was used for these tests with no use of companding or similar compression algorithms. Companding (and other compression algorithms) do exist to reduce quantisation noise for a given bit representation and can reduce the bandwidth requirement. These algorithms may form an important part of a more complex headset design. However, the exact choice, development and use of such algorithms were not considered within the scope of this project.

During data transmission, channel issues can cause bit errors within packets and/or corruption of entire packets resulting in distorted or missing audio information. While the human auditory system can certainly overcome audio gaps and distortion to some degree - for example 20mS of corrupted audio is not serious enough to be noticeable to the human ear (Truax 1999) - the bit error rate (BER) and packet error rate (PER) must be kept to a minimum to maintain audio quality. The exact effect a certain BER and PER have on the audio quality largely depend on the digitisation and compression technique, with higher compression algorithms generally suffering to a greater degree. For an example, Figure 2-6 shows the Perceptual Evaluation of Speech-Quality (PESQ) MOS versus BER for a 12.2 kbps CELP. (The figure also displays BER curves for link strategies that retain packets with bit errors in the least, middle and most significant bytes).

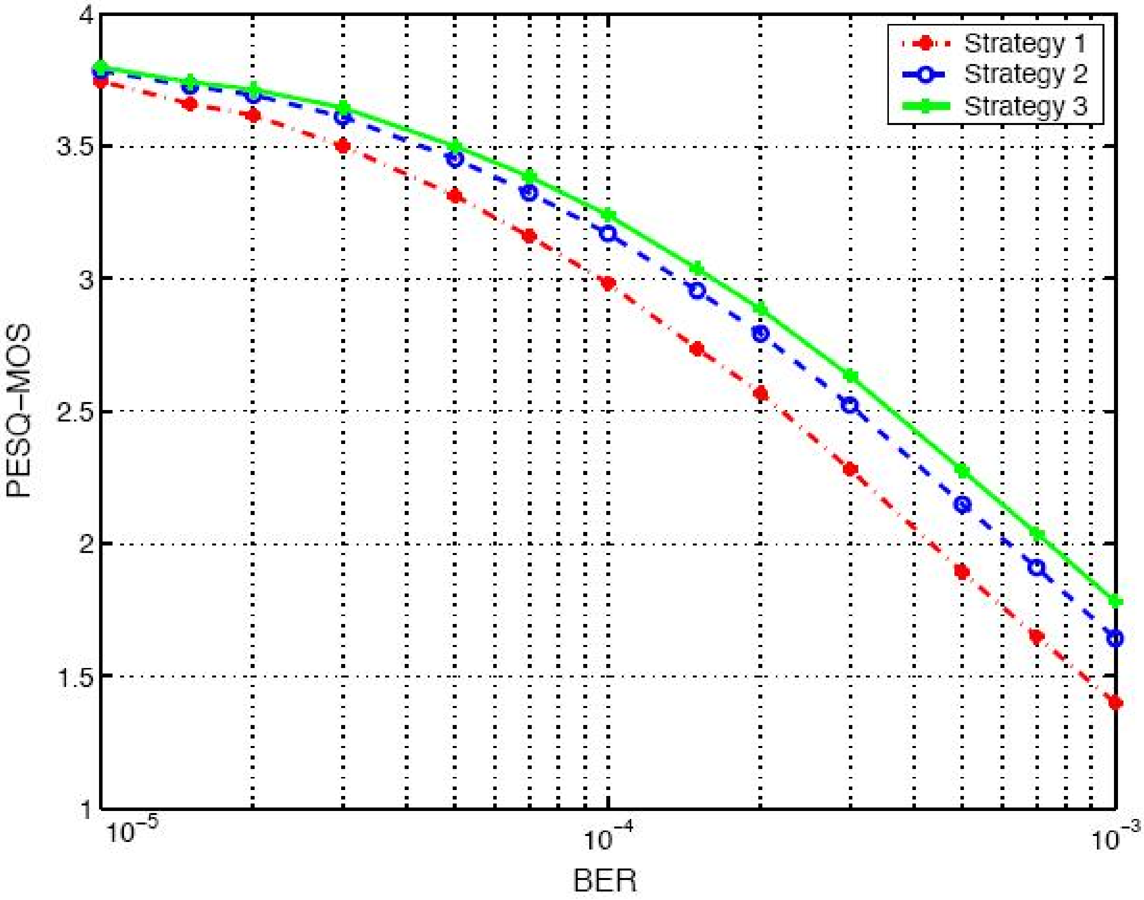


Figure 2-6: Estimated perceived speech-quality vs. bit error rate (Hammer 2003)

The reduction in MOS for speech intelligibility and resultant hearing fatigue must be considered for VIPs using the headset for extended periods as long a working day.

2.2.3.4 Delay of computer to headphone audio

Given that VIPs need to audibly scroll through menus, lists and other GUI type layouts with screen reading software, it is important to keep the delay from a PC to headphones as low as possible. For example, a VIP scrolling through large amounts of data quickly will be affected in their ability to correlate the focus of the PC pointer relative to the spoken text if the delay is too large.

A headphone system must consider the total delay accumulated from audio compression, transmission from PC to headphone and subsequent decompression. Whilst a comprehensive study has not been performed on the allowable delay for the speech to text applications, the International Telecommunications Union specifies a maximum end to end delay of no more than 150ms for acceptable telephony voice quality (Cisco 2006).

As with many real time communications digital transmission, the re-sending of corrupted data packets is not practical in this application. This influences the allowable packet error rate of the system, and enforces consideration for forward error correction and audio gap filling techniques such as noise insertion and waveform filling.

This criterion is particularly relevant to the choice of a speech compression algorithm, as they typically exhibit an inverse relationship between delay and compression effectiveness.

2.2.4 Security

Call centres often deal with sensitive information such as credit card numbers and personal information; hence it is vitally important that the confidentiality of that data is not undermined. Thus the communications between the information source (computer) and the VIP call centre employee should not expose that data, or allow it to be intercepted by unauthorised persons.

Regardless if wireless headsets are utilised or not, call centres are still subject to vulnerabilities such as internet based hacking, untrustworthy employees and security of the PSTN with which the call centre interacts. The security implications of using a wireless headset communications system are that the communications system must be;

- as secure as practically possible so that the added risk of using a wireless headset system does not add to the vulnerabilities a call centre may already be exposed to;
- manageable to the point where it should follow the security provided by the physical environment (walls, doors, floors, ceilings and windows).

2.2.5 Robust and scalable operation

For work environments, such as call centres, the operation of a wireless headset needs to be;

- reliable, in that the quality of the audio needs to remain at the required level regardless of internal or external interference noise;
- scalable, in that the quality of the audio needs to remain at the required level regardless of number of users of the system.

Also, the headset design must comply with standards for preventing acoustic shock syndrome (ACS). ACS is caused by sudden or unexpected noise events and results in hearing trauma at sound pressure levels well below those which present a risk of immediate hearing damage. This is due to the increased fragility of hearing when using a headset for extended periods of time. (NPL 2006)

2.2.6 Ergonomic considerations

For work environments such as call centres, a wireless headset needs to be;

- of acceptable weight if it is going to be worn for a standard eight hour shift;
- practical size, so that the user should find wearing the headset comfortable, not limiting their natural range of movement and not embarrass them due to its size, shape or form;
- able to operate for a full shift on a single battery charge as any charging during the shift could inconvenience the user and the call centre.

These requirements present a battery capacity (mAh) versus. weight compromise that can only be overcome by minimising the power consumption of the system, even considering state of the art lithium ion type batteries.

2.3 Research into the headset design problem

Four undergraduate projects at Curtin University of Technology have attempted to create a viable solution to this headset problem for VIPs.

Dodds (1996) developed an analogue infrared system where stereo audio was transmitted to a headset with AM modulation for one channel, and FM modulation for the other. According to Dallavolta (2001) a working prototype was developed. However problems encountered with the system were as follows;

- *"Audio level changes in the AM channel, when the distance from the transmitter was changed.*
- *High frequency audio interference when placed near a computer monitor.*
- *Only single directional transmission was available, that is, no microphone was incorporated for user feedback to the PC.*
- *Inability of power management in the remote unit due to analogue nature of the receiver."*

DallaVolta (2001) redesigned the headset to use digital infrared communications (with the IrDA standard) in combination with a digital signal processor (DSP) for control and in many ways laid the groundwork for this thesis. His work also more clearly defined the basic requirements of the system and developed a controlling algorithm.

Todd (2002) continued the work of DallaVolta with better defined requirements. He also examined the key influences speech compression and control hardware have on the system. His work also highlighted the need for considering the turn-around time characteristic of duplex infrared communication. A working system was not developed.

Crake (2002) investigated a microcontroller based infrared headset and developed a software solution for a low performance microcontroller in combination with IrDA type transceivers. A working system was not developed but his work suggested that a DSP based system would be more suitable due to the performance requirements.

Significant academic and commercial research has been conducted into radio communications for wide area networks (especially for cellular mobile telephony and mobile LAN) and Personal Area Networks (for example Bluetooth). WAN communications research is dominated by the needs of larger cells and sophisticated equipment and is not ideal in areas that would serve the requirements of a wireless headset for VIPs. However the substantial research and development being conducted into PANs suggest that in due course they may meet the requirements of VIPs.

3.0 An assessment of communication concepts in application to the VIP headset problem

3.1 Communication requirements and media, modulation and multiplexing options

To find the most suitable communications solution to any problem, it is first necessary to profile the target environment. The primary target environment being considered here is a call centre with VIP employees. That sets some significant constraints. In particular, the following:

- The close proximity of users in an office environment. An example layout of a typical call centre is portrayed in Figure 3-1. Call centres, though, can vary greatly in terms of the density of users and geometrical layout.

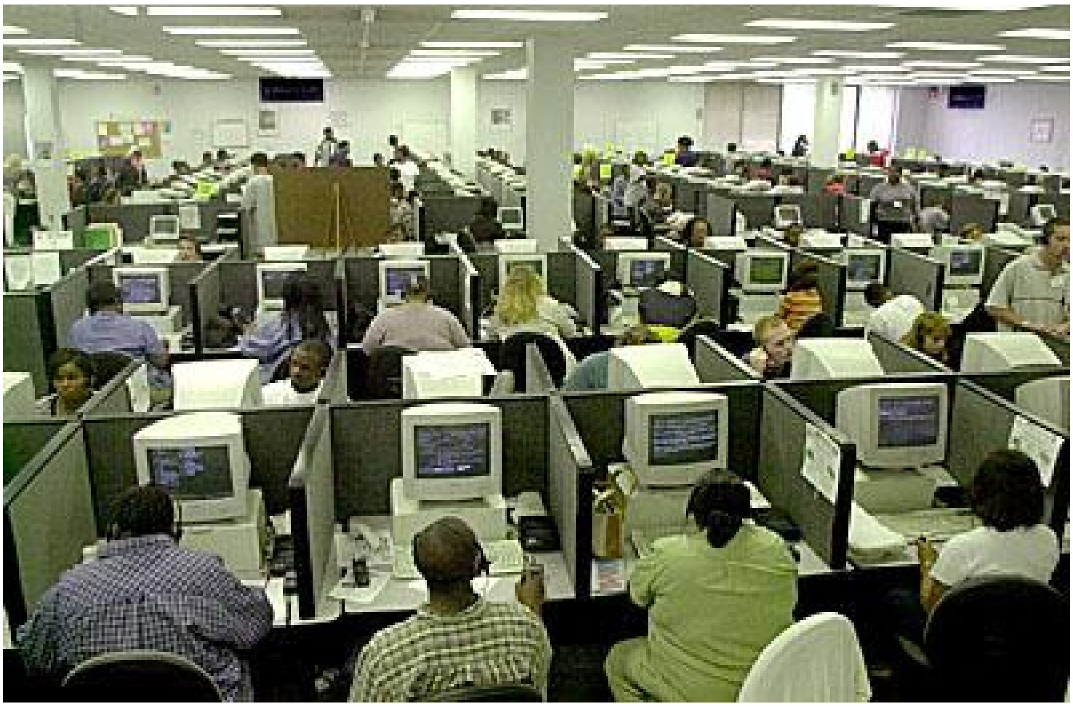


Figure 3-1: High Density Call Centre (Hickey, M 2008)

- Semi or fully closed cubicles partitioned with a variety of office type materials such as fibreboard, metal and plastic composite. Situations do exist where there may be no partitions, but this is typically in the case of low density call centre environments where adjacent user acoustic interference is not a problem.

- A high likelihood that the call centre is within proximity to other businesses, and possibly located within a multi-story building.
- An asymmetry in bandwidth required for base-to-headset and headset-to-base communications. For example, if the stereo headset was to use PCM coding with telephony voice at 8khz, 8-bit representation and speech to text audio at 16KHz, 16 bit representation then,

Base-to-headset bandwidth = (speech to text audio bandwidth) + (telephony voice bandwidth)

= (16000 * 16) + (8000 * 8)

= 256kbps + 64kbps

= 320kbps.

Headset-to-base bandwidth = generic voice bandwidth

= 8000 * 8

= 64kbps

This represents an asymmetric link of 5 to 1.

The type of voice compression and other bandwidth saving techniques used, such as Voice Activity Detection (VAD), will alter this ratio to some degree and that has to be factored into the final design.

Possible communications media that can be used in this situation are as follows;

- Radio frequency electromagnetic communication. Within the spectrum there are many sub-bands that have unique characteristics in terms of path loss, reflection co-efficient, power requirements and other parameters.
- Infrared communication.
- Acoustic/Ultrasonic communication.
- Magnetic field communication.

For each of the communication media there is also several ways to communicate data. Factors to be considered include the following:

- *The modulation of information onto the medium*. Given the need for high quality audio, security and a controllable noise level, only digital transmission techniques need be considered. Options include Phase-shift keying (PSK), Frequency-shift keying (FSK), Amplitude-shift keying (ASK), Quadrature amplitude modulation (QAM) - a combination of PSK and ASK, Gaussian minimum-shift keying (GMSK), Orthogonal frequency division multiplexing (OFDM) modulation, Wavelet modulation and Trellis coded modulation (TCM) amongst others. (Wikipedia 2008c)
- *The media access control (multiplexing) technique*. This can be either a circuit switched or packet mode method. Given the real time requirements of voice and its sensitivity to delay and jitter, a circuit switched connection is usually better provided it can be implemented. Common circuit switched (channelization methods) and packet switched options include Time-division multiple access (TDMA), Frequency division multiple access (FDMA), Orthogonal frequency division multiple access (OFDMA), Wavelength division multiple access (WDMA), Direct-sequence spread spectrum

(DSSS), Frequency-hopping spread spectrum (FHSS), Orthogonal Frequency-Hopping Multiple Access (OFHMA), Carrier Sense Multiple Access (CSMA), Code division multiple access (CDMA) - the overarching form of DS-SS and FH-SS amongst others. (Wikipedia 2008d)

The modulation and multiplexing techniques each have their own and combined set of qualities in terms of;

- bandwidth efficiency;
- resilience to noise;
- complexity of implementation.

that needed to be considered when applied to the chosen communication medium.

3.2 Analysis of communication media

3.2.1 Radio communications

3.2.1.1 Sound quality / bandwidth, scalability and reliability

3.2.1.1.1 One base per headset implementation: a personal area network topology

To implement wireless headset communications for all users in a given environment, one solution would be to have a one base-station per wireless headset system where the telephony and speech to text audio can be fed to the base-station straight from the VIPs local work area. Given the limitations of the radio network this generally would limit the user's ability to stray from their station without causing interference to other users. However this roaming functionality is not required within a call centre. A detailed examination of such a PAN wireless communication system suggests FDMA and FHSS are probably the best techniques to use.

The UltraWideBand (UWB) standard also shows great promise in delivering the capabilities required for this headset design due to its high bandwidth, reliable and ease of implementation characteristics. However at time of writing the UWB standard, IEEE 802.15.3a draft PAN, has been dissolved due to industry deadlock on implementation (Haasz J 2006).

The implementation of a low frequency (50MHz – 1GHz) FDMA type system within a call centre environment would require at least two independent frequencies per user for duplex operation. Hence a high number of independent frequencies would be required as only a low level of frequency re-use would be possible with the high density of users. Further, a rigid setup and maintenance program would also be needed to ensure inter-user interference remains at acceptable levels. As an analogue example of an FDMA system, VHF FM (Very High Frequency, Frequency Modulation) headsets are a simple and cheap type of wireless headset; an example is pictured in Figure 3-2. This example model can only support four users at once and operates in the unlicensed 72–76 MHz band which is also occupied by radio controlled models and industrial remote control communications. VHF headsets are unsuitable for VIPs in call centre environments as they are insecure, have low capacity (in terms of number of users supported), and are prone to interference.

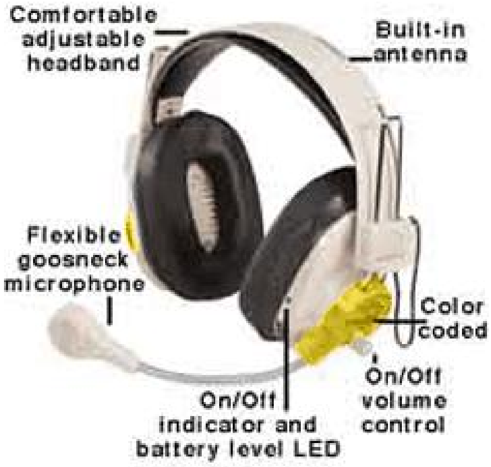

Comfortable adjustable headband
Built-in antenna
Flexible goosneck microphone
Color coded
On/Off volume control
On/Off indicator and battery level LED

Figure 3-2: Califone FDMA Headset (AudioLinkServices 2003)

It is plausible that a high frequency (10GHz-100GHz) radio system could allow a simple FDMA solution. High frequency radio has the benefit of being high bandwidth, highly directional and with favourable propagation loss and limited penetration characteristics. It can also operate without direct line-of-sight (LoS). Such characteristics enable a highly cellular network to be easily implemented while also allowing acceptable freedom of movement for the user. The developing WirelessHD standard (refer to http://www.wirelesshd.org/technology.html) presents a suitable commercial solution. However at time of writing it has not been ratified and commercial products are not available.

Frequency Hopping Spread Spectrum (FHSS) transmits radio signals by rapidly switching a frequency carrier among many channels and offers the following advantages over an FDMA type system (Wikipedia 2008a):

- *Spread-spectrum signals are highly resistant to narrowband interference. The process of re-collecting a spread signal spreads out the interfering signal, causing it to recede into the background.*
- *Spread-spectrum signals are difficult to intercept. An FHSS signal simply appears as an increase in the background noise to a narrowband receiver. An eavesdropper would only be able to intercept the transmission if they knew the pseudorandom sequence.*
- *Spread-spectrum transmissions can share a frequency band with many types of conventional transmissions with minimal interference. The spread-spectrum signals add minimal noise to the narrow-frequency communications, and vice versa. As a result, bandwidth can be utilized more efficiently.*

Bluetooth, or IEEE 802.15, is a prevailing FHSS standard amongst PAN commercial devices and has the following features (Bluetooth 2007):

- Frequency of operation from 2.402 – 2.480MHz (unlicensed spectrum).
- 79 channels, 1MHz wide each.
- A hopping frequency of 1600 hops/s.

- GFSK modulation with 1 Ms/s symbol rate, or 3 Ms/s with new specification EDR (enhanced data rate).
- Three radio transmission power classes.

Power Class	Maximum Output Power (Pmax)	Nominal Output Power	Minimum Output Power*	Power Control
1	100 mW (20 dBm)	N/A	1 mW (0 dBm)	Pmin<+4 dBm to Pmax Optional: Pmin** to Pmax
2	2.5 mW (4 dBm)	1 mW (0 dBm)	0.25 mW (-6 dBm)	Optional: Pmin** to Pmax
3	1 mW (0 dBm)	N/A	N/A	Optional: Pmin2** to Pmax

- Master – slave relationship.
- Time division duplex operation (TDD).

Two types of data links are possible in the Bluetooth standard; SCO (Synchronous Connection-Oriented) for real time data such as audio/video and ACL (Asynchronous Connection-Less) for non-real time, high bandwidth data. They have the following characteristics:

- SCO
  - Point-to-point connection between master and slave. The master maintains the link by using reserved timeslots at regular intervals. Packet retransmissions are not allowed.

  - 64 kbps audio data rate utilising continuously variable slope delta modulation (CVSD) coding, a compression scheme that can handle frequent bit errors in operation.
  - Up to 3 bi-directional SCO links are possible per master.
- ACL
  - Provides packet-switched connections between the master and all active slaves. Packet retransmissions are usually applied to assure data integrity.
  - Allows up 433 kbps symmetric or 723.2 kbps asymmetric data rates.

There are several packet types for each of these connections as shown in Figure 3.5

Table 1: Properties of Packet Types, Zurbes et al (2000)

Link type	Packet type	Payload FEC code rate	User payload – [Bytes]	Burst length [µs]	Occupied slots
Control	NULL		0	126	1
	POLL		0	126	1
ACL	DM1	2/3	0 – 17	171 – 366	1
	DM3	2/3	0 – 121	186 – 1626	3
	DM5	2/3	0 - 224	186 – 2871	5
	DH1	1	0 - 27	150 -366	1
	DH3	1	0 - 183	158 – 1622	3
	DH5	1	0 - 339	158 – 2870	5
SCO	HV1	1/3	10	366	1
	HV2	2/3	20	366	1
	HV3	1	30	366	1

A typical example of a FHSS Bluetooth stereo headset with microphone return is the Logitech Mobile Stereo Headset HS 200, pictured in Figure 3-3.

Figure 3-3: Bluetooth Stereo Headset

Given a call centre application has an asymmetry in bandwidth requirements for base to headset and headset to base communications; this suggests the following setup may be used;

- Base-to-headset communications (requiring a maximum of 320kbps for uncompressed PCM audio):
  - High bit rate ACL channel for speech to text audio. According to Xue et al (2001), the high bandwidth ACL data channel can be used to support high bit rate audio. Due consideration must be given however for the greater delay, jitter and decreased reliability of the link compared to the SCO channels.
  - SCO connection for telephony stream.
- SCO connection for the headset-to-base telephony communications (~ 64kbps in uncompressed PCM form).

From an examination of Bluetooth’s properties, it is quite possible that the standard can provide the necessary bandwidth to meet the VIP headset requirements in a call centre environment. However the impact of congestion and interference on the reliability and system noise level of Bluetooth audio arising from bit and packet errors is of concern.

Zurbes et al (2000) found that for a 10m x 20m room with randomly placed master and slave piconets, Bluetooth exhibits high frame erasure rates (FER) and residual bit error rates (RBER) for the three forms of SCO link with large numbers of concurrent sessions. Their results are summarised in Figure 3-4.

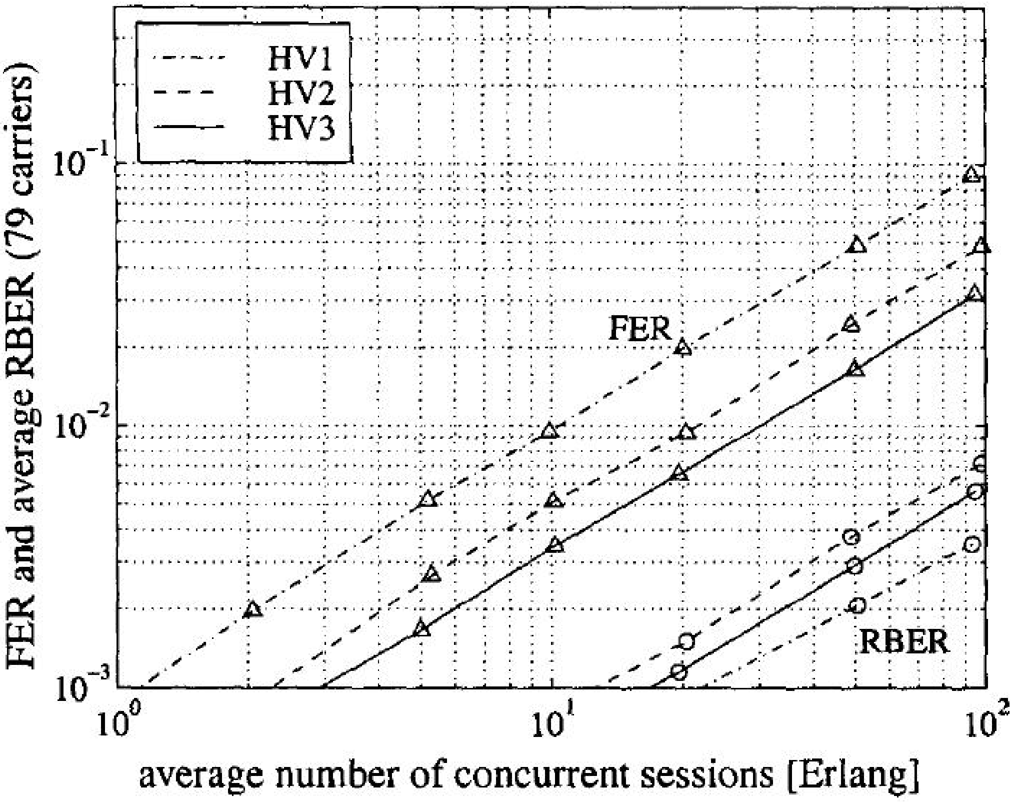


Figure 3-4: SCO Link Quality Measures (79 carriers), Zurbes et al (2000)

Howitt (2003) developed an analytical model, supported by empirical tests, to determine the maximum base to mobile distance spacing (d_S) to maintain a desired packet error rate with varying degrees of adjacent user activity (probability Pr[A]), user density (D_I) and signal path loss exponents (n). Figure 3-5 depicts the relationships required to maintain a maximum 5% packet collision rate. Given the probability of adjacent user activity within a call centre this may be 100% for extended periods of time when users are on call, user densities of $1/2^2$ as well as $1/5^2$ when the n = 2, will require a d_s of less than 1.5m. This is reasonable for an immobile user operating a computer terminal. However when considering the headset audio quality requires a PER of less than 10^-4 (and not 5%), it is likely that user density will have to be very low (less than $1/10^2$), and this would exclude most call environments

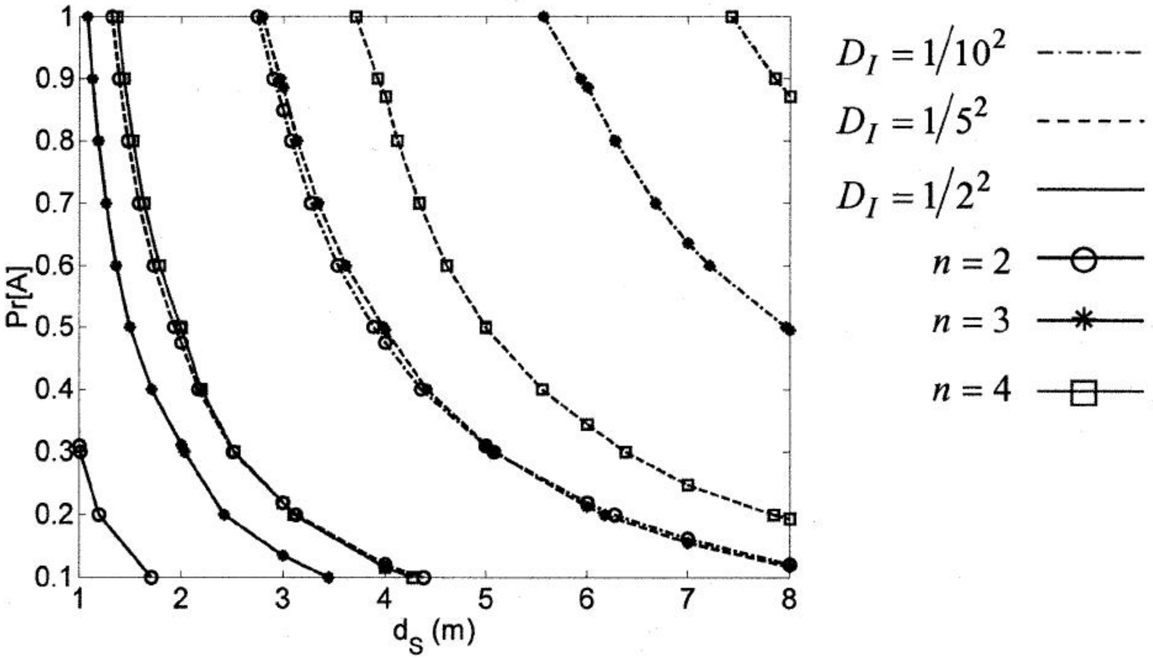


Figure 3-5: Packet Collision Rate Curves, Howitt (2003)

Bluetooth’s performance when subject to narrowband interference can be considered generally very robust due to its FHSS protocol. However broad spectrum noise, such as a 22MHz wide Wifi channel, can significantly impact the PER and BER of a Bluetooth link. Zhengzhong et al (2004) documented that when Bluetooth is subjected to varying 802.11b packet rates, high bit error and frame erasure rates result, as shown in Table 3.1 below.

Table 2: Results of 802.11b interference on voice quality Zhengahong et al (2004)

Test	i	ii	iii
Input Speech	In1.wav	In2.wav	In3.wav
Packet Rate	200	500	999(Always On)
AWGN	Off	Off	OFF
Hop Frequency	Random	Random	Random
BER	1.6e-2	4.6e-2	9.2e-2
FER	3.8e-2	0.12	0.21
Output Speech	Out4.wav	Out5.wav	Out6.wav

Audible effects of this interference include “cracks, pops and dropouts” (Merritt, et al2004) and Nour et al (2004) also found that in relation to the more robust SCO link;

> *“Bluetooth packet loss rates may reach up to 38% in unfavourable 802.11 interference conditions, and as Bluetooth uses a CVSD codec with syllabic companding, these packet losses not only manifest themselves as segments of missing speech upon CVSD decoding, but also as incorrect scaling of subsequent successfully received voice packets as CVSD step-size information is also lost.”*

This performance is not an acceptable for the SCO telephony link and hence it is unlikely that the standard Bluetooth protocol and commercial equipment can provide the necessary reliability and robustness for use in a call centre headset.

There are however numerous (non-standard) measures that can be employed, often in combination, to improve Bluetooth performance. These include the following:

- The use of a directional antenna on the base station to direct the Bluetooth radio wave to the headset in a 120 degree angle to the user. This will greatly reduce the rate of co-channel and adjacent channel interference (given the asymmetric link) while also maintaining freedom of movement ergonomics for the user in the cubicle.

- Efficient power control schemes for Bluetooth transmission power classes 2 or 3;
    - Power control is only required in the Bluetooth standard for class 1 devices. It uses a RSSI (received signal strength indicator) parameter to control the transmitting power of the paired device. Given the high density of users in a call centre, an efficient power control scheme for class 2/3 will reduce congestion and improve reliability. For example, considering that most of the time the user is in LoS with the base only low power Tx will be required for most situations and higher transmitter power is needed if the signal is shadowed by user movement in the cell.
- Synchronising Bluetooth piconets to reduce PER,
    - The synchronizing of piconets allows for a twofold increase in the maximum achievable throughput (Ashraf et al 2006)
- Tailoring the environment to the technology. For example by,
    - Spreading the distribution of users in the call centre to decrease adjacent user interference
    - Removing sources of interference such as Wifi and microwave ovens.

If the listed measures above are not possible or sufficient, the purchasing of licensed spectrum may be the only reliable solution. Licensed spectrum, however, adds significant cost, especially if an equivalent 79MHz of bandwidth is required. Also obtaining 79MHz of vacant spectrum in an appropriate frequency band may be extremely difficult.

3.2.1.1.2 One base - multiple headset implementation: a wide area network topology

As an alternative to a personal area network, a wide area network topology allows finer control and coordination of signals between adjacent users and permits more efficient use of bandwidth resources. Also, by virtue of it's greater and uniform coverage the WAN topology also allows improved roaming functionality if it is desired.

When considering WAN wireless communications protocols, the most viable commercial solutions at present for a call centre problem were found to be TDMA and OFDM implementations. CDMA protocols (such as CDMA2000 and UMTS) are highly suitable for delivering wireless voice communications, but power consumption and difficulties surrounding CDMA patents negate the protocol for this particular scenario.

When implementing a wide area network for a wireless headset system, there are two means of communicating the PC and telephone audio to the headset and each has their own advantages/disadvantages;

- Via a wired local area network (LAN) to a wireless access point: This will allow efficient use of wireless bandwidth resources. However, it will also require the LAN infrastructure to have QoS protocols implemented or necessitate a separate "voice only" LAN.

- Or wirelessly to an access point: This will halve the capacity of the system given the need to transmit the same information twice (as defined in most WAN protocols such as IEEE 802.11 for signal control and roaming purposes). However no changes to LAN infrastructure will be required.

Given the difficulty in providing reliable wireless communications, the first option was considered the better option in this case.

TDMA protocols for wireless voice communications are common and widely developed. DECT (Digital Enhanced Cordless Telecommunications) is a common TDMA implementation that also combines advanced multi-carrier protocols (FDMA) and can operate in the 900MHz, 1.8GHz, 1.9GHz, 2.4GHz or 5GHz frequencies, depending on the country. An example of a DECT headset that nearly meets the requirements of VIPs in call centres is the Plantronics CS361N shown in Figure 3-6 below.

Figure 3-6: Plantronics Stereo DECT Headset (Plantronics 2006)

DECT standard headsets are unsuitable for VIPS in call centre environments (Kowalk 2005) due to;

- low quality audio based on 32kbps ADPCM (Adaptive Differential Pulse Code Modulation);
- weak encryption which permits vulnerable communication;
- susceptibility to congestion and other sources of interference.

More advanced forms of TDMA protocols do exist to address these problems, such as the GSM suite of protocols. However in general the TDMA protocol is inferior to OFDMA in terms of link efficiency and resilience to noise, and thus was not considered.

OFDM (Orthogonal Frequency-Division Multiplexing) uses a spread spectrum protocol that distributes the link data over a large number of carriers that are spaced apart at precise frequencies. This spacing creates orthogonal frequencies, meaning the peak of one sub-carrier coincides with the null of an adjacent sub-carrier and this prevents radio demodulators from seeing frequencies other than their own. The benefits of OFDM are high spectral efficiency, resiliency to RF interference, and lower multi-path distortion (Wave Report 2007) and it is used in many common wireless protocols including IEEE 802.11a/g/n.

IEEE 802.11a/g/n standards have significant industry backing with matured, commercially available products and were investigated to determine if they could meet the requirements of the wireless headset application. As an example, the 802.11a standard features 22MHz wide channels, CSMA/CD for multiple access, a base rate of 54 Mbps and a real data rate of about 20Mbps. 802.11a/g/n is however tailored to the delivery of non-real time data and is quite poor in efficiency for delivering voice (Wang et al 2006). For example, 802.11a can only support approximately 25 uncompressed, 64 Kbit/s PCM voice streams per channel in an ideal situation (Waclawsky et al 2004) and is particularly inefficient when compared with GSM, CDMA in terms of call capacity per MHz bandwidth.

There are however various methods proposed, including new standards such as IEEE 802.11e and 802.11n which use QoS and beam forming (via "multiple in, multiple out" (MIMO) antennas) techniques respectively which can greatly improve the call capacity per access point. There are also many non-standard methods of optimising channel capacity. In the methods proposed by Wang et al 2006 to reduce packet header and MAC overheads, when accepting a packet loss rate of 1% and using a GSM 6.10 codec (~13kbps) the call capacity can be increased greatly from 27 with standard 802.11E to 245 when optimized. (Refer to Figure 3-7)

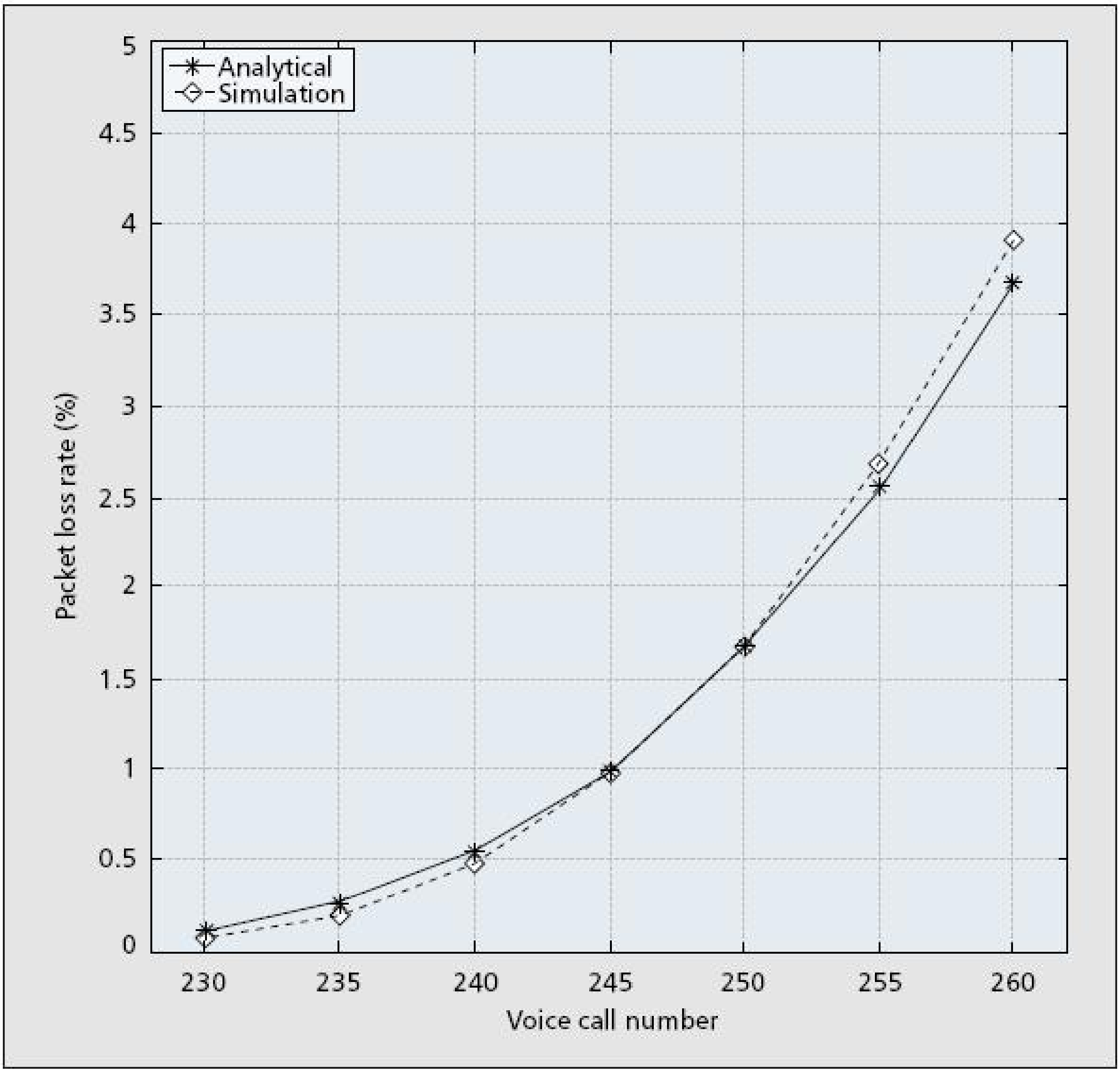


Figure 3-7: Packet loss rate for 802.11e following optimisation

The VIP wireless headset needs for low PER, dual channels and high quality audio create more demanding requirements than those considered by Wang et al 2006) and Waclawsky et al 2004. However, it is quite possible that at least 25 headsets could be accommodated with a single 802.11a channel. Furthermore, to accommodate a greater number of users for call centres with 25 plus users, a 802.11a cellular network could be implemented with a 1-7 frequency re-use scheme if appropriately chosen directional (downward pointing) antennas and network planning tools are used.

There is concern however regarding the reliability of such a network when subject to interference from other 802.11 WANs on the same frequency or other wireless protocols using the unlicensed bands (Linden 2003). The reliability of 802.11a/g/n in an unlicensed spectrum is a major concern unless sources of noise can be removed and proximity to other businesses using the same protocol avoided. The ideal solution, as it was for FHSS, would be the use of licensed spectrum. Licensed spectrum, as stated previously, will however add significant cost and additional operational overhead to setup and maintain.

3.2.1.2 Security

Radio frequency electromagnetic fields are characterised by long range signal propagation and the ability to penetrate through common building materials such as masonry and glass/steel structures. Naturally this means that any radio communications made within a call centre can be easily picked up outside the physical perimeter causing a security concern for institutions such as banks, governments and defense forces (Xydis and Wilson, 2002). While wire mesh wall paper, metallic (lead based) paint and specially tinted windows can be employed to contain the radio signal, these methods are costly and are not completely effective.

To protect the radio signal information, encryption protocols such as AES can be implemented that may provide adequate security for most environments. For situations however where absolute security is required, a One-Time-Pad (OTP) system would need to be implemented (Spectrum IEEE, 2003). One-Time-Pad systems use a stored cipher key that can be used only once, and whose length is as long as the data to be transmitted. For a full day’s operation with PCM audio, the

One-Time-Pad solution would then require both the base and headset to have a non-volatile memory capacity of,

$256_{\text{Kbps(speech to text)}} + 64_{\text{Kbps(telephony)}}\ 64_{\text{Kbps(microphone return)}})\ \text{x}\ 60_{\text{secs}}\ \text{x}\ 60_{\text{mins}}\ \text{x}\ 8_{\text{hrs}}$

= 11.06 Gb

= 1.38 GB.

2GB flash storage chips are commercially available, low power, compact, inexpensive and could easily be accommodated into compact headset. A key passing mechanism would need to be employed, and this could be performed during non business hours, possibly by a wired link. This method of information security will achieve the necessary data protection, but will require extra user intervention to ensure secure communications.

3.2.1.3 Ergonomics

The non-LOS characteristic of radio frequency wave propagation allows unrestrained freedom of movement for the VIP user.

In terms of headset size and weight, radio electronics can be very compact and light. However, in a radio headset design, the heaviest component will be the battery. Bluetooth protocol hardware can be very low power (a feature of the standard) and, for example, draws only 12mA from a 1.5V supply for a 12Kbps HV3 voice link (Cojocaru 2005). The higher bandwidth required for the VIP headset will likely consume a multiple of this. However, with VAD and other schemes battery size and weight should not be a major concern. The same cannot be said though for commercial OFDM hardware, which due to the large number of signal carriers and computational requirements are typically very power hungry and result in the need for an impractical large battery for 8 hours of operation. At present, no dual channel

OFDM headsets exists to give an exact indication of real life power consumption. However there are numerous single channel versions and examples include;

- Dlink DPH-540 Wi-Fi Phone, which with all lower power options enabled can only last 3 hours with a rather large 1100maH battery according to DLink 2007.
- Netgear WiFi Phone with Skype SPH200W, which according to Netgear 2007 can only last 4 hours with a 900maH battery

Thus, battery size and weight is a major stumbling block for an OFDM headset until battery and DSP technological progress can provide a reasonable solution. As a complete alternative, the battery could be body mounted, or swapped out with a spare when depleted, but this may introduce the tangling problems for VIP users that the wireless headset design is trying to avoid.

3.2.1.4 Possible concerns regarding radiation safety

One widely debated yet unproven concern regarding the use of radio communications is the health and safety from close range, all day exposure to radiation (BBC2007). Even though the radiated power of 802.11 is only 0.1W, a twentieth of the power of a mobile phone (max~ 2W), the use of an OFDM headset for an eight hour working shift may raise safety issues with employers and employees. Further study may need to be undertaken to determine the risk level associated with all day exposure.

3.2.2 Acoustic / ultrasonic communications

3.2.2.1 Sound quality / bandwidth

Research into acoustic communications, particularly for underwater applications where ultrasonic frequencies exhibit excellent transmission characteristics with bandwidths reaching 120kbps (Ochi et al 2008) is extensive. However for atmospheric environments, the physical medium conditions greatly limit the achievable bandwidth with acoustic communications. Holm (2005) reports a usable bandwidth of only 100bps, and using advanced spread spectrum techniques IBM (2000) reports a possible bandwidth of 3.4kbps. This data rate is too low to support the required wireless headset channels and audio fidelity.

3.2.2.2 Security

Acoustic communications exhibit good security characteristics in that the signal propagation closely follows the physical security provided by the environment, and actually can be considered more secure than the vocal propagation of the wearer's own voice (due to the higher space attenuation of ultrasound).

3.2.2.3 Ergonomics

High frequency ultrasound can has a working range of up to six metres. However the narrow propagation beam width and inability to penetrate common materials limits user movement restrictions. Ultrasound, however, does reflect well off hard surfaces this might be used constructively to help maintain the data link.

Ultrasonic transducers are small, light and low power, and the size and weight of a wireless ultrasonic headset would be reasonable.

3.2.3 Nearfield Magnetic Communication (NFMC)

3.2.3.1 Overview

NFMCs utilise magnetic induction for the transfer of information. The propagation of the signal follows the magnetic lines as shown below in Figure 3-8.

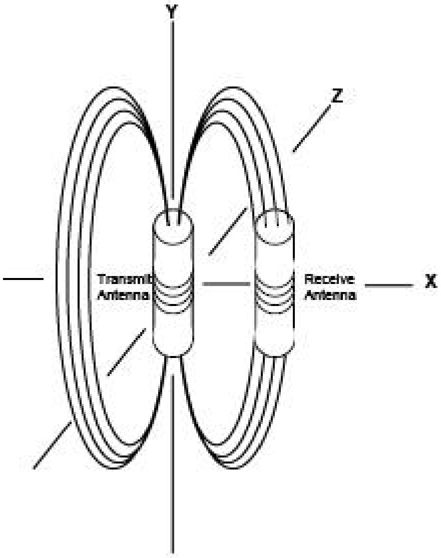


Figure 3-8: NFMC Coaxial Orientation (Auracomm 2003)

One advantage of this type of communications is that the signal attenuates in free space with a 1/radius^6 characteristic, as compared to radio waves having an attenuation of 1/radius^2 as depicted in Figure 3-9. This allows a higher density of users per given volume with low congestion and also robust security due to the considerable attenuation of the signal at distance.

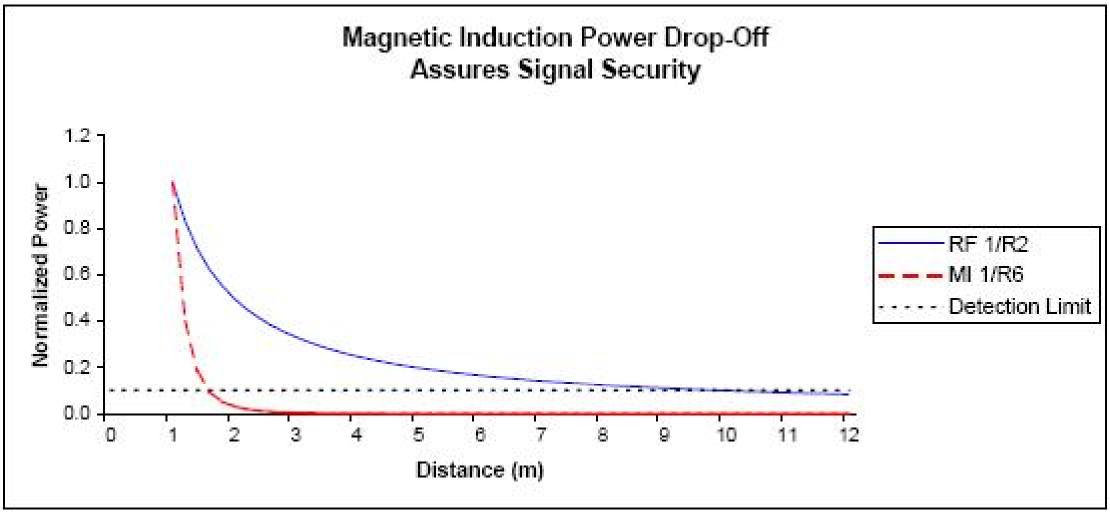


Figure 3-9: Magnetic Induction Signal Attenuation with Distance (Auracomm 2003a)

Significant developments in NFMC have occurred recently in this field. However the technology is proprietary (Aura Communications, 2003a). A commercial example is the Auracomm Docker headset, depicted in Figure 3-10.

Figure 3-10: Docker Magnetic Headset (MobileBurn 2003)

3.2.3.2 Sound quality / bandwidth

At the time of writing, commercial NFMC transceiver hardware can support a raw baseband bandwidth of 204kbps, which is then time division duplexed. With communication protocol overheads and turnaround time implications (to transition between transmit to receive states), this only allows a 64kbps audio data rate. As a result, commercial headset implementations are single channel only and exhibit a poor audio frequency response of 30Hz to 3.8KHz (Auracomm 2003b). This does not meet the required dual channel, 8KHz frequency response for high speed voice applications and thus would need to be significantly improved if it is to provide a suitable platform for VIPs.

Fully re-usable frequencies are possible every few meters (Auracomm 2003b) and that permits the density and scalability required in a headset. It also offers an easy implementation path. Unfortunately no BER data is available on NFMC or its operation in congested environments.

Commercial NFMCs operate in the 13.56MHz band that very few devices share. However the progressive deployment of RFID systems that also use the 13.56MHz band, may present future interference problems and so there might need to be a condition that RFID devices are not co-located with NFMC headsets (Gratton 2007). Also the use of CVSD coding in commercial audio implementations (Figure 3-10 for example) suggests design has to consider a noisy and high BER communications channel as CVSD is typically chosen for such purposes.

3.2.3.3 Security

NFMCs are inherently very secure due to the advantageous 1/R^6 signal attenuation. However magnetic signals can propagate through masonry and glass materials commonly used in building structures and as shown in Figure 3-9, there is still measurable signal power at around 2m. Thus some consideration must still be given to;

- the placement of users close to the perimeter of a workplace;

- call centres in multistorey buildings as it may be possible for eavesdroppers to pick up the signal from the ceiling of a below level.

3.2.3.4 Ergonomics

The non-LOS characteristic of NFMC allows good freedom of movement for the VIP user. The size and weight of a NFMC headset can also be very small, as demonstrated with commercial examples, as the transceiver hardware typically has very low power consumption thus allowing a small and light battery. For example, Auracomm (2003c) claims "*less than 15 mA at 2 V is required to transmit full-duplex continuous voice or data across a 1.25-m link, dropping to 10 mA at 0.75 m. This is the total current, including ASIC current, transmitter power, speaker power, and microphone bias."* Hence, combined with a small 260mAh Li-Ion battery, 15 hours of battery life may be possible which is more than sufficient for the intended application (Gratton 2007).

3.2.4 Infrared/optical communications

3.2.4.1 Overview

Headsets can utilise the infrared electromagnetic spectrum (~300THz) for the transfer of information and its characteristics exhibit many similar qualities to that of visible light such as limited boundary penetration and high available bandwidth. There are many infrared headphone sets commercially available for consumer applications such as listening to music, but only one commercially developed headset at present with bidirectional voice transfer, the IR-Link IR100 pictured below in Figure 3-11.

Figure 3-11: IR-Link IR100 Wireless Infrared Headset (IRLink 2004)

Requirements specifications for this type of wireless headset would typically include the following: (IRLink 2004)

- Transmission Method: Full Duplex Wireless Infrared, Digital transmission.
- Ni-MH Rechargeable Battery.
- Operating Time: 6 hours of continuous use in duplex mode.
- Operating Range: 3m.
- Alert sound warning when out of range.
- Receive/Transmit volume control.
- Mute button.
- Power Consumption;
    - headset : 25mA/3.6V;
    - base station : AC/DC Adapter, 90 mA/9V.

Such headset features are good solutions to meet the needs of VIPs in call centre environments. However the IR100 was monophonic and production has discontinued.

3.2.4.2 *Sound quality / bandwidth*

The two major forms of IR communication are Diffuse (non directed, non LoS) and Directed LoS.

Diffuse IR is typically used in wireless LAN deployments requiring the interconnection of many users. It utilises a combination of high-power (long-range) signals and wide transmit (Tx) and receive (Rx) angles to allow reflected signals to maintain communications when an object blocks the direct LOS. This allows reasonable user freedom of movement in operation. However the same reflections may cause multipath distortion which severely limits its bandwidth (Ramirez 1999). Also the wide Tx and Rx angles reduce receiver sensitivity and create great difficulty in implementing a cellular architecture in a open office environment such as a call centre, and thus multiple access techniques (such as CDMA or CSMA) must be used which will further deplete the already limited link budget. Jivkova (2003) suggests using a multi-spot diffusing (MSD), MIMO architecture to improve this situation and reduce the probability of blockage. However power efficiency remains a concern during shadowing and link robustness is still questionable. If the office setup makes use of enclosed cubicles/rooms, and users are not sitting back to back, then adjacent user interference will not exist, and many of the listed problems disappear. However, this type of setup is rare.

Directed LoS infrared communications are clearly designed for links with only direct LoS and typically have very low power, high bandwidth and narrow Tx and Rx angles which allow much greater power efficiency than diffuse IR (Ramirez 1999). Also given the low power and narrow Tx and Rx angles, direct LoS suffers low adjacent user interference and naturally lends itself to the implementation of a cellular network. Hence, in most cases, the need for multiple access schemes are not required, and in the unlikely scenario that it is, it can be easily accommodated in the healthy link budget. The major disadvantage of directed LoS IR however is that it is very prone to dropouts from Tx to Rx due to angular misalignment and shadowing which will occur from the user's body movements (in single Tx and Rx arrangements). While there are numerous techniques for reducing the angular and shadowing limitations of direct LoS IR, these add complexity and cost issues which generally limit their application. Typical examples include the following;

- *Spatially Diverse Transceivers (SDT*). Multiple transmitters can be used to allow more than one link path to the receiver and thus help to maintain the link in case of blocking or pitch, yaw or lateral movements, and may be applied to both or either the base and headset. In a call centre application, a scenario can be envisaged where base transceivers are placed either side of a computer terminal to overcome headset blockage or angular misalignment issues. However the use of SDTs may lead to increased IR noise within the call centre environment and hence greater probability of interference.
- *Angle Diversity Transceivers (ADT).* 360 degree coverage could be achieved with multiple transceivers to compensate for the limited viewing angle of individual transceivers. This implementation will require non receiving transceivers to power off transmission if the user rotates their headset away

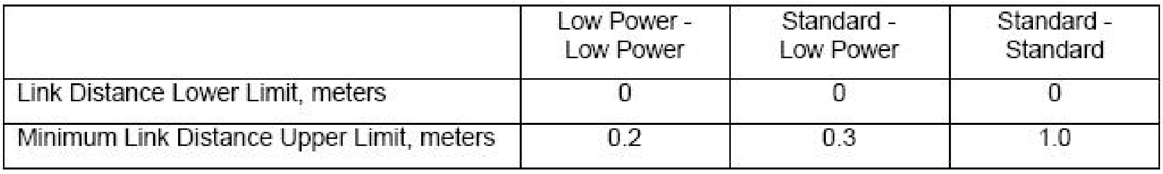
from the base to limit adjacent user interference and power consumption. This technique brings increased complexity in the form of physical implementation and also increased computational requirements, as the data from all transceivers needs decoding. The problem of LoS blocking also cannot avoided.

- *Self Orientating Transceivers (SOT).* If a suitable mechanical tracking system is implemented to maintain LoS regardless of user angular and lateral movements then the limited viewing angle limitation of a single transceiver system can be circumvented (Castillo-Vazquez et al 2003). This can be considered a single transceiver variant of ADT with reduced computational requirements. The problem of blocking is still not avoided however and the ability to maintain LoS with fast user movements needs to be considered. It also is also expensive to manufacture and mechanical reliability is a concern.

The predominant standard for directed LoS IR communications is the Infrared Data Association (IrDA) standard. Many low power, miniaturised components are available to construct systems to conform to it. The IrDA Data standard has the following specifications; (IrDA 2001):

- Baseband data rate of 9.6kbps to 16Mbps.
- Modulation: RZI (Return-to-Zero-Inverted) for 9.6kpbs to 1.15Mbps and PPM (Pulse Position Modulation) for 4Mbps and above.
- A specified data link maximum BER of 10^{-8} at 1m operation.
- Half duplex operation (full duplex communication is not feasible since when transmitting a device’s receiver is blinded by the light of its own transmitter).
- Power levels for operation distance defined in Table 3.

Table 3: Link Distance Specification (IrDA 2001)



	Low Power - Low Power	Standard - Low Power	Standard - Standard
Link Distance Lower Limit, meters	0	0	0
Minimum Link Distance Upper Limit, meters	0.2	0.3	1.0

- Minimum receiver half angle of 15° and transmit half angle of 15° to 30° as depicted in Figure 3-12 below.

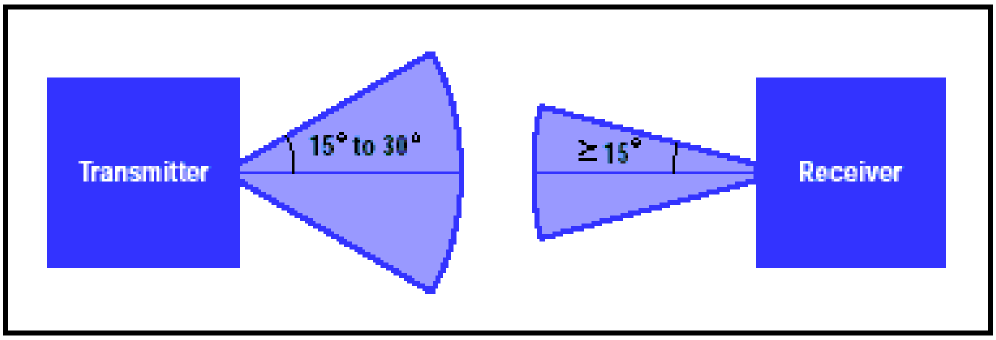


Figure 3-12: IrDA viewing angle for receiver-transmitter pair. (Millar 1998)

The tight Tx and Rx angles aid the implementation of a cellular network with a high signal-to-noise-ratio (SNR) within the call centre environment, and make the IrDA standard a practical choice for the wireless headset. Barker (1999) provides an analysis of the effect distance and transmitter angle has on BER for a given transmitter cosine lobe exponent (m), and the profile plot can be seen in Figure 3-14. A receiver lobe index (m) of 20 is indicative of the minimum IrDA specification of ± 15° receiver half angle. (Barker 2002)

~

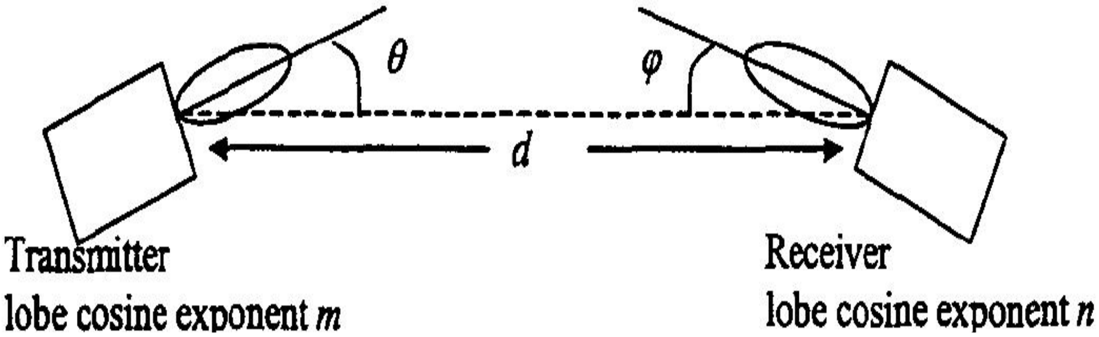


Figure 3-13: IR Link Topology (Barker 1999)

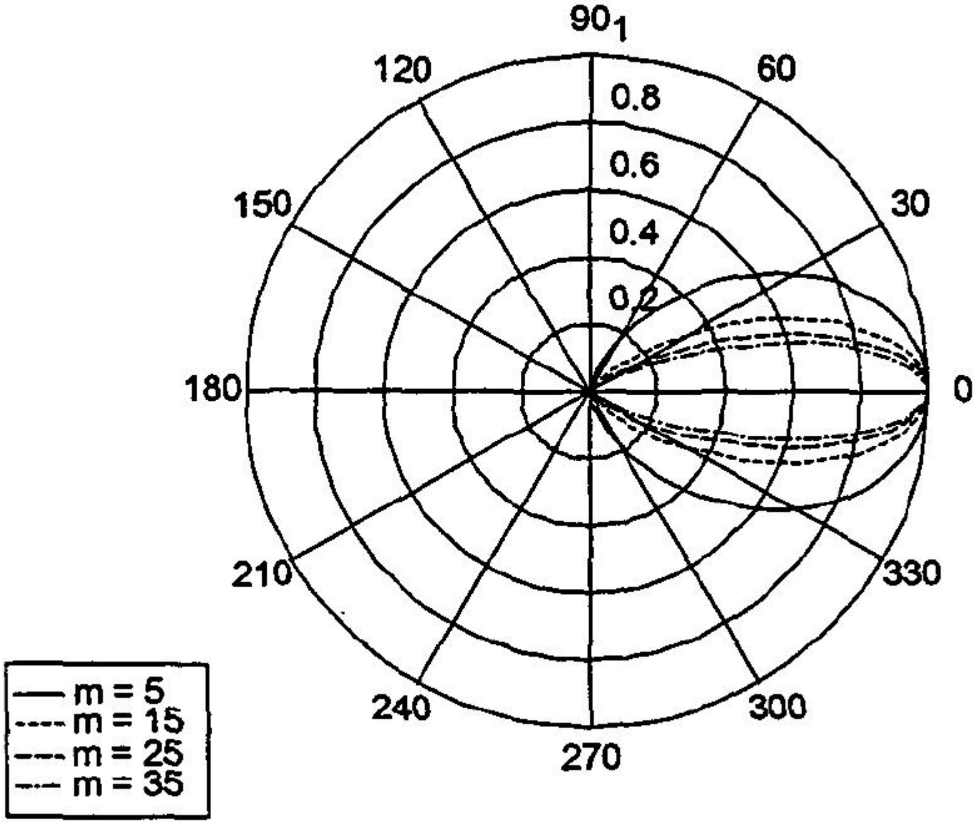


Figure 3-14: Profile plot of BER Vs distance and angle for IrDA Link (Barker 1999)

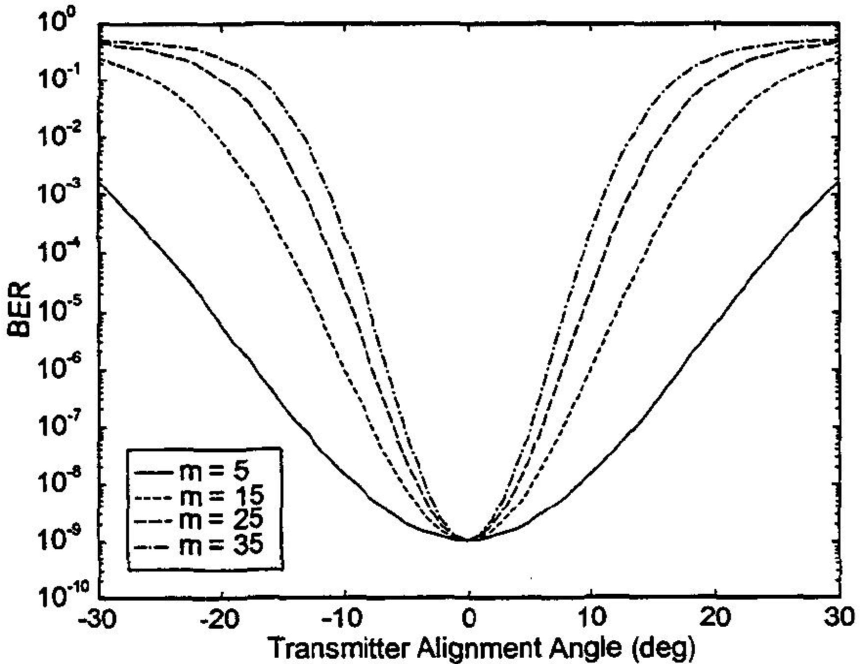


Figure 3-15: BER Vs Tx Angle for IrDA link with d = 1 meter (Barker 1999)

Barker (2002) also analysed the effect an interfering IrDA user has on the BER of an IR link with varying distances and angles, as depicted in Figure 3-16.

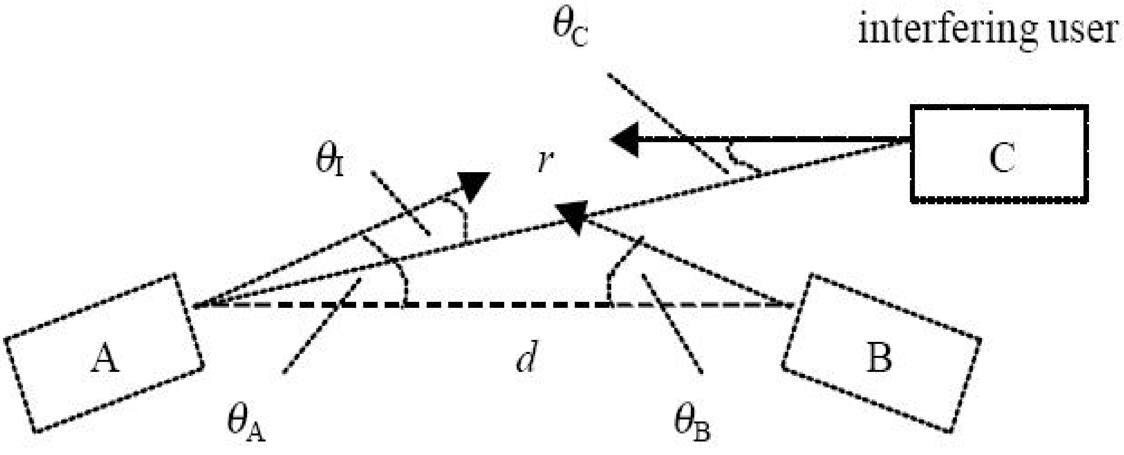


Figure 3-16: Geometry of link affected by third user interference (Barker 2002)

Figure 3-17 provides a plot of the minimum interferer approach distance (r) and alignment angle (θ_I) to provide link BER values of 10^{-8} and 10^{-7}.

(Assuming d = 1m, and $\theta_A = \theta_B = \theta_C = 0°$)

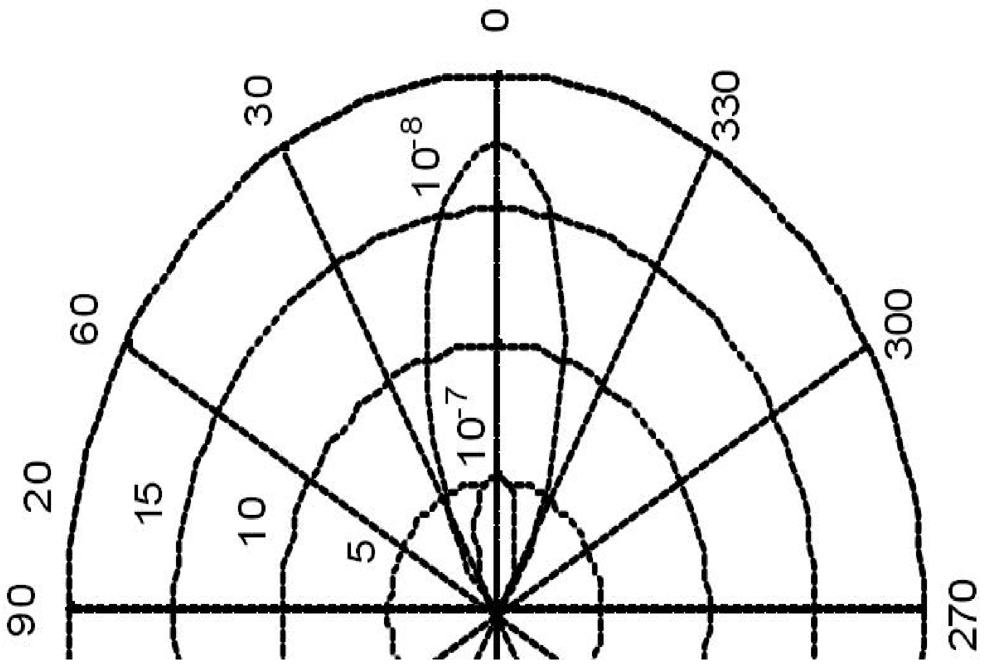


Figure 3-17: Polar plot of interferer distance (m) and angle to provide link BER of 10^{-8} and 10^{-7} (Barker 2002)

Figure 3-17 shows that in an uncontrolled environment without 360-degree enclosing partitions an interferer directly behind the user needs to be at least 5m away to maintain a BER of 10^{-7}. However given the maximum BER for high quality audio needs to be 10^{-4} for the headset design, the separation distance must be shorter. This is illustrated in Figure 3-1 and Figure 3-18 and is indicative of the worst case scenario.

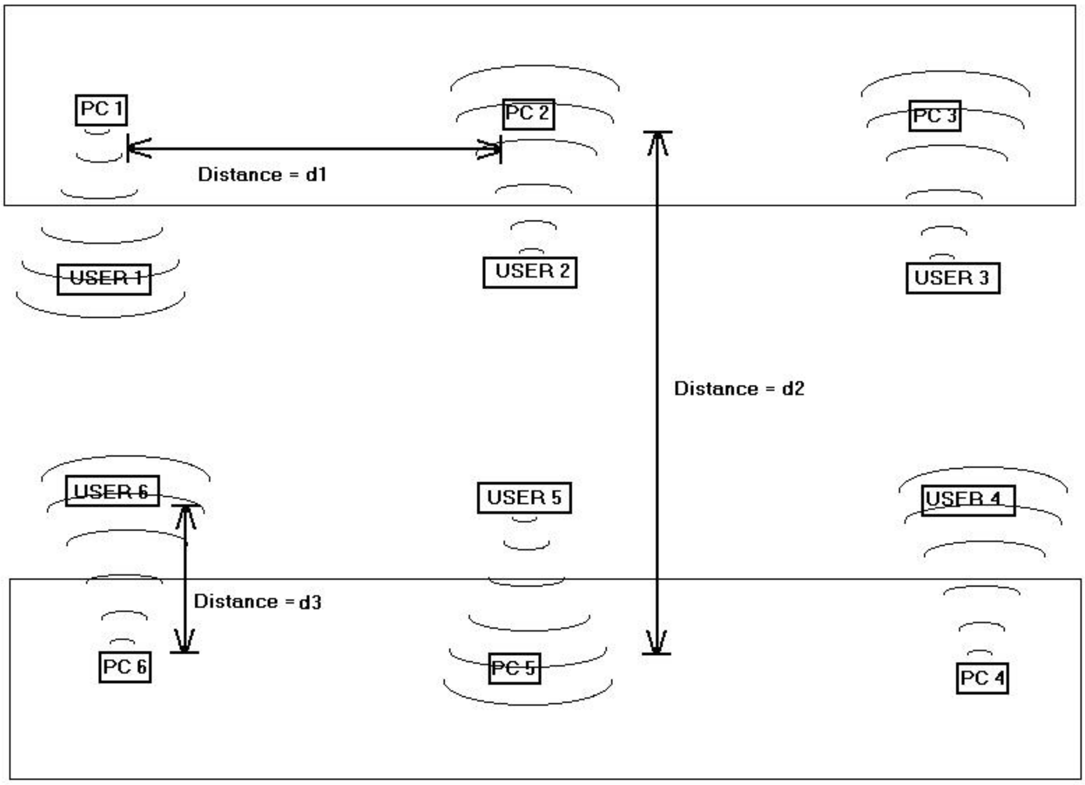


Figure 3-18: Top View of the Worst case layout Transmission (DallaVolta 2001)

Adjacent user interference (e.g. User 2 to User 1 or 3) is not considered an issue as it is assumed that headsets have basic power control and do not transmit IR when they are not in LoS and hence communicating with their paired base station.

The separation distance could be shortened however if the limited transmit and receive field of view for directed LoS IR is taken advantage of and an angle exists between the users. Figure 3-19 shows a layout in a typical high density call centre environment and in this case User 1 - User 5 now becomes the worst-case interference combination. Here, the minimum distance for d2 reduces to $\sqrt{}(d2^2 – d1^2)$, and although this does not greatly reduce the back to back distance, if the partition walls extend far enough behind the users to block potential interference between User 1 and User 5 base stations, then a cellular architecture could be easily achieved.

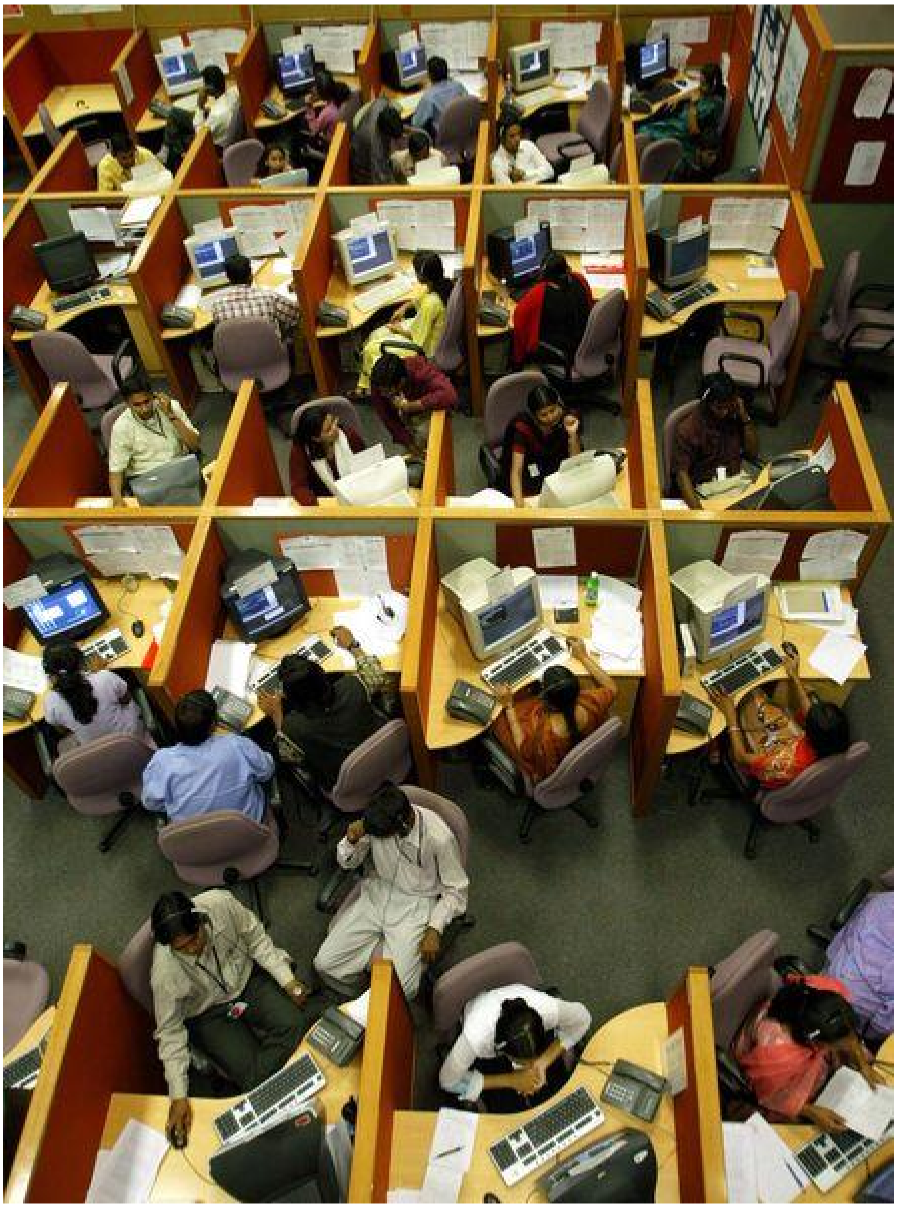


Figure 3-19: Typical High Density Call Centre (ABCNews 2007)

Hence with the use of partitions a cellular network may be implemented with IrDA headsets within a call centre environment, thus allowing high bandwidth, low noise communications and the required sound quality. The primary limitation is the need to maintain constant LoS for reliable communications and the associated unrealistic impositions it places on user freedom of movement.

3.2.4.3 Security

Infrared signals are easily blocked by walls and other non-transparent materials in a similar manner to visible light and thus the security of communications can easily be contained to the physical work environment. In terms of glass openings, most forms of window glass are designed to reflect IR energy (especially if tinted) and hence the ability of an eavesdropper to pick up the signal outside of a glass window is greatly reduced. Furthermore, the addition of window curtains can be used to eliminate the vulnerability altogether. Hence infrared communications are considered in general to be an extremely secure form of wireless communications.

3.2.4.4 Ergonomics

The LoS characteristic of directed LoS infrared greatly limits the user's freedom of movement when using a headset and it is unlikely that full day use could be comfortable or ergonomic. Non-LoS IR can reduce this problem to some degree. However, as stated previously, the performance of this form of IR is not suitable in many regards.

Infrared communications, especially short range directed LoS forms, can have extremely low transmission power and also require relatively low computational resources to control. This allows the use of a small and light battery for all day usage. The size of the transceivers are also miniaturised and so an infrared headset is quite ergonomic in size and weight.

Figure 3-20: Typical IrDA Transceivers (Myslik 2003)

3.3 Communication media assessment summary

For the development of a wireless headset in a call centre environment, a summarised assessment of the various communications media and techniques discussed is as follows;

- RF communications have many positive ergonomic features and are highly developed. Further methods exist to alleviate security concerns. However the reliability of communications in the presence of interference is a major issue that cannot be overcome easily or cheaply without licensed spectrum, and that brings significant cost.
- NFMCs have excellent characteristics in terms of cellularity and security. However there are concerns regarding;
    - intellectual property implications in developing a customised version to suit a call centre headset;

    - whether 64kbps (data rate) is sufficient to provide suitable quality audio for stereo channel operation;
    - how the BER will be affected with users in closer proximity;
    - how reliable heavily compressed audio (such as CELP) will be in providing high quality audio under these circumstances;
    - minor security implications for call centres located in multistorey buildings.
- Infrared provides a good combination of bandwidth, security, and power consumption. It is also highly robust when in LoS. However it is unlikely that a person using the headset would easily maintain the correct pitch, yaw and lateral position throughout a working shift. While the use of SDTs, ADR and SOR will improve the ease of maintaining LoS, they add complexity and are limited in their effectiveness and thus a directed LoS infrared system is too unergonomic for this use.

3.4 Solution proposal: IR/RF hybrid concept

No one communications medium easily satisfies all of the requirements of the wireless headset. However, a combination of media can. The previous discussion suggests a particular hybrid namely directed LoS IR (such as IrDA) to carry the base-load of the communication along with a non LoS RF PAN (such as Bluetooth) acting as a link backup. The idea of a hybrid infrared and radio system is not new and there has been considerable research surrounding the combination of the two. For example;

- Hou et al (2006) focuses on the efficient and reliable handover between IR and RF and highlights the "*complementary nature both in capacity and coverage motivates us to consider using both for data transmission in an* *integrated system. The potential benefits of such a system are decreased delay and hence, better quality of service (QoS)*";

- Sakurai et al (2003) describes the advantages of the hybrid system and how to functionally achieve seamless handover over an IP network between optical and RF NICs (network interface cards);
- Tsai et al (2004) suggested a novel method of modulating a RF signal onto an IR carrier to benefit from the cellular high bandwidth characteristics of IR when it is in LoS.

Thus to cater for the requirements of a wireless headset for VIPs in call centre environments, the following conceptual solution was developed:

- The IR link carries the base-load of the audio data, providing high quality, interference free operation.
- The RF PAN (Bluetooth) carries the link when the user turns or walks away from the desk and in doing so greatly improve the ergonomics of the headset:
  - The RF PAN may be subject to lower quality audio under interference. However this may not be significant as the user is unlikely to be operating their computer and thus not requiring high quality audio. Furthermore if there is severe interference in the area, the user can simply reface their terminal (and base station) to regain the high quality audio they require.
  - With the IR link providing the base-load of communications, there is a reduced probability of FHSS packet collisions from adjacent users and hence it is unlikely that the RF PAN will experience significant congestion based interference. Also the impact on the link BER/FER

when FHSS packet collisions do occur is less severe as there is a low possibility that adjacent user are operating under the backup RF PAN, and the increased probabilistic spatial diversity will lessen the impact to the link SNR.

- To maintain security of transmission, OTP keys will be swapped between base and headset by the IR link and stored in a buffer for later use when operating with the radio link. The memory requirements of this system are quite low. For example, if a high bit rate audio link of 256kbps is used, then for the headset to be able to communicate securely for 5 minutes without LoS, then it would require memory capacity for the storage of OTP keys of size:

$$256000 \text{ x } 60 \text{ x } 5 / 8 = 9.6 \text{ MB.}$$

, which can be easily and cheaply implemented in RAM or FLASH storage.

There will also be a requirement to provide some indication to the user that they have lost IR communications. This feature will ensure that the user does not deplete the OTP key buffer and compromise the RF security. An unobtrusive warning tone indicating the occurrence of this situation is a possible solution.

4.0 The suitability of IR for base-load wireless communications in an IR/RF hybrid headset

4.1 Introduction

To prove the IR/RF hybrid concept it was necessary to test whether IR could practically and robustly provide the base-load of wireless communications for the headset. The ability of IR to reduce RF airtime to a minimum and thus avoid RF congestion and the debilitating impact from external interference was critical to the concept's success.

The concept uncertainties could not be proven easily or accurately by theory or simulation; hence a test bed was designed and developed. The test bed was used in laboratory and actual call centre environments, and definitive data of the performance of the IR/RF concept in real life conditions has been obtained. The test results reveal key information regarding the strengths, limitations and necessary compromises of the hybrid concept.

4.2 Tests required to prove the IR/RF hybrid concept

4.2.1 The effectiveness and ergonomics of IR in providing base load wireless communications

For directed LoS IR the angular range of operation raises a direct compromise in terms of how ergonomic the headset will be in use and how robust a cellular architecture can be implemented for a call centre environment. For example, increasing the angular range of operation for the IR transceiver system will improve the probability of LoS communications and hence the ergonomics of the headset, but there will be greater probability of interference between users.

Using standard IrDA transceiver optics as a benchmark, Figure 3-14 gives an indication of the BER and expected link quality with varying degrees of transceiver movement. However real life testing was needed to identify how normal VIP movements of pitch, yaw and lateral position will actually influence the effectiveness of directed LoS IR in providing the base-load of wireless headset communications. The influence of user training and adaptation (which is aided by link quality feedback) also needed to be considered along with possible ergonomic issues (stiff neck for example) that may arise if users trying to maintain LoS communication adopt unnatural postures.

Furthermore, the real life effectiveness and ergonomics needed to be tested with realistic headset configurations (and respective transceiver locations). Examples included;

- headset with transceivers mounted on headset earpieces (both sides);
- headset with a transceiver mounted in shirt pocket.

4.2.2 The robustness of cellular IR communications in call centre environments

Using standard IrDA transceivers, Figure 3-17 gives an indication of how spatial orientation can influence inter-user interference and the BER of an IR link. However the ability to practically implement a cellular architecture with realistic headset designs within a call centre environment and the compromises that must be made were unknown and required investigation. The effect of user separation distance, geometrical layout and cubicle partitions were considered.

In gaining this design data, the following was assumed:

- A shared infrared frequency range based on identical transceivers.
- Headset transceivers have simple power control and do not transmit when not in communication with their paired base. This may be implemented using a unique base ID number in the communication header frame and will greatly limit spurious infrared transmissions.

4.3 Infrared headset test bed

4.3.1 Test bed overview

The headset test bed, shown in Figure 4-1, is designed specifically to the requirements of a VIPs working in a call centre environment, but it may also be easily adapted and applicable to other scenarios. It is described as follows: (Todd 2002)

- The system consists of two units. One has continuous power and acts as a 'base station'. The other unit is portable, battery powered and wearable by a test subject as a headset.
- The Base Station takes audio input from two sources consisting of a telephony channel (from a PSTN handset or VOIP system) and a text to speech synthesized voice channel from a computer. This is relayed wirelessly to the headset:

  - The incoming telephony channel is reproduced at the headset without loss of audio quality and meets the standard of the PSTN system being an 8-bit companded, 8 KHz signal.
  - The speech to text synthesized voice channel is reproduced with 16-bit, 16 KHz PCM or equivalent compressed audio quality without noticeable delay.
- The Headset wirelessly receives the telephony and synthesized voice from the Base Station and output each independently to the left and right speakers on the Headset.
- The Headset take audios input from one source - a microphone on the unit. This is relayed to the Base Station.
- The Base Station wirelessly receives the test subject's microphone audio from the Headset and outputs it to the telephony system (PSTN handset or VOIP system).

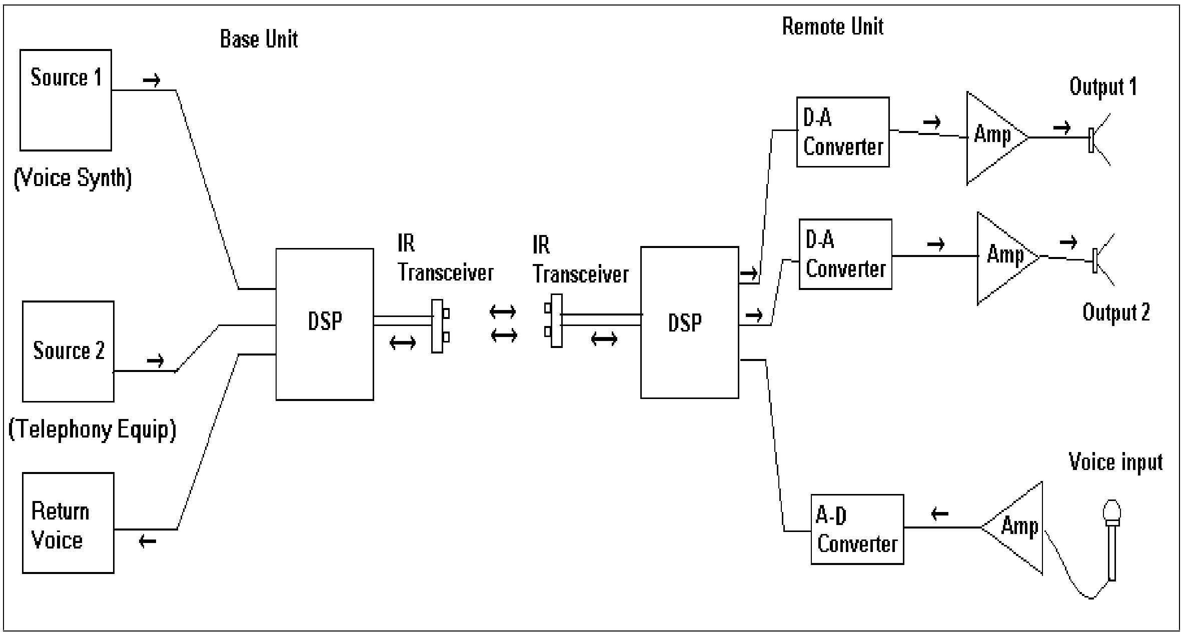


Figure 4-1: Basic System Overview (DallaVolta 2001)

The exact choice of hardware for the test bed considered the headset's requirements for low power and fast data manipulation and also the flexibility required of a test bed undergoing testing, reviewing and adaptation cycles. If required this flexibility would allow the implementation of advanced techniques such as SDT, ADT or SOT to improve ergonomics, multiple access schemes (for example CDMA, CSMA) if infrared cellularity wasn't possible as well many other base/headset combinations (single-base/multiple headset, multiple base/single headset for example) if the initial test design proved unsatisfactory.

A design centered on the use a digital signal processor (DSP) combined with commercial IrDA transceivers provides a practical solution to the functional requirements. DSPs are efficient communications processors and provide the flexibility to change and re-engineer communications protocols and respective characteristics quickly and easily via software. Also with numerous IO ports available, the general peripheral hardware used with the DSP (analogue audio interface, system memory (Flash) and IR transceiver type for example) can be easily interchanged and adapted. Specifically, a Texas Instruments TMS320VC5502 DSP was chosen for its high speed (24-200MHz), low power and low cost features.

The Agilent HSDL-3602 was chosen as the IrDA infrared transceiver for the test bed. It has the following features: (Agilent, 2003a)

- Data transmission and receive rates of up to 4Mbps
- Distance of operation up to 1.5m

- 3 power levels to choose from (33%, 66% and 100%) which may allow reduced power consumption and improved IR cellularity via BER based power control schemes
- Small surface mount form factor.
- Transmitter “stuck on” protection.
- Adaptive Threshold Control (ATC): The IR quantiser threshold for the receiver logic ‘high’ is dynamically adjusted according to the average of the incoming signal amplitude. This is to allow sensitivity to low power transmissions whilst also preventing low power noise from triggering a false logic ‘high’ in stronger powered signals.

The transceivers are mounted on small PCBs (see appendix III.r) and utilise bendable wire mounts to allow transceiver angles to be tailored to the wearer’s requirements.

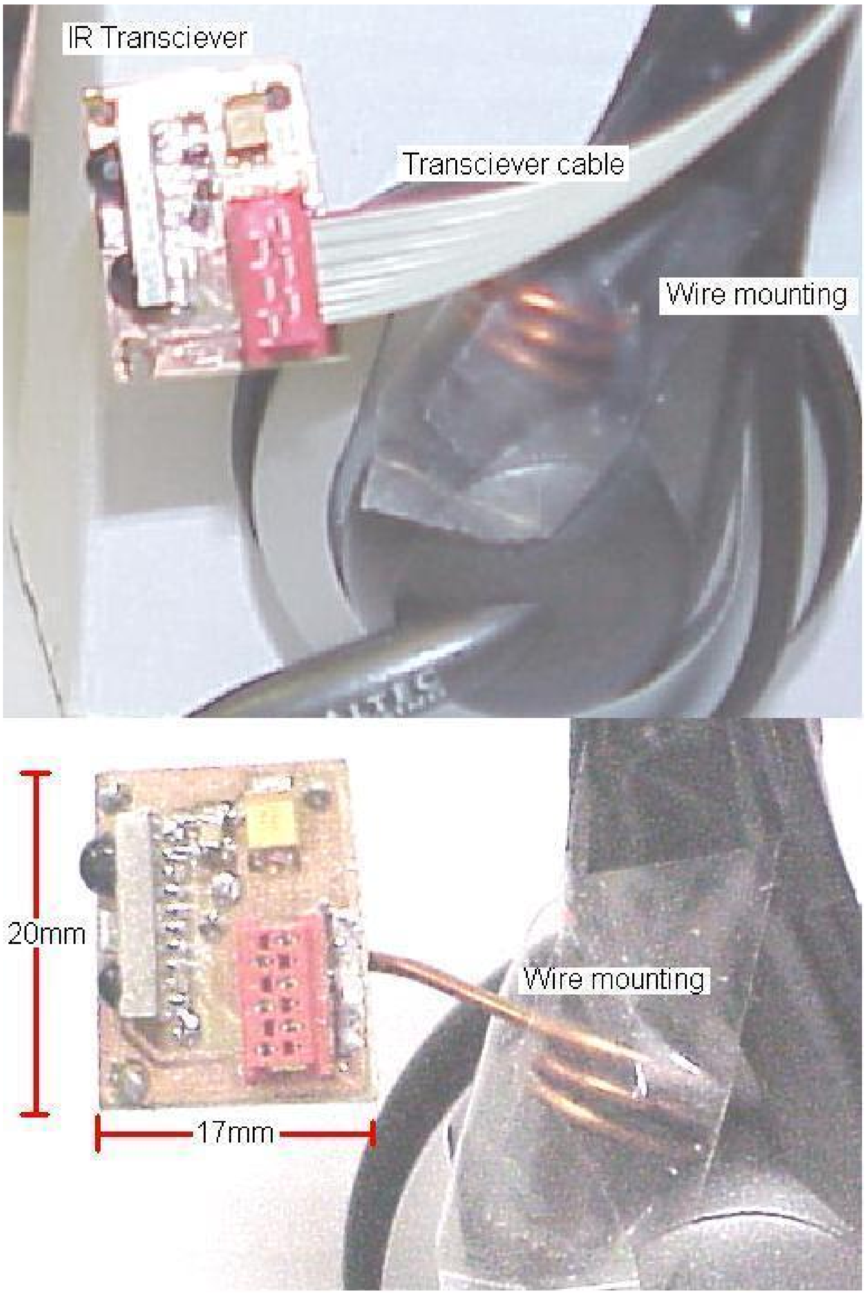


Figure 4-2: Infrared Transceiver Mounting

4.3.2 Base station prototype

The prototype base station utilises a single transceiver design. However additional transceivers can be easily added (to aid headset ergonomics) if required. The hardware architecture is depicted below in Figure 4-3 and includes a charging system for the headset battery.

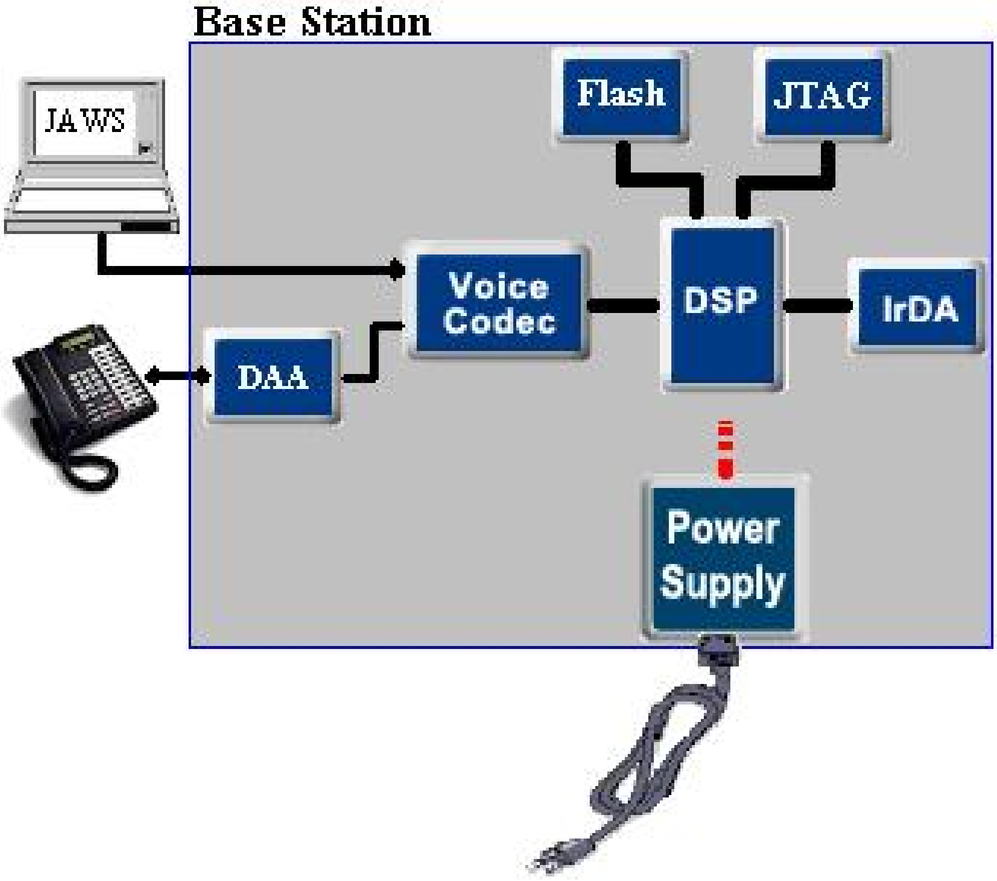


Figure 4-3: Base Station Hardware Overview

Appendix III.m reveals a close up of the actual base station PCB circuitry and the base station layout is show below.

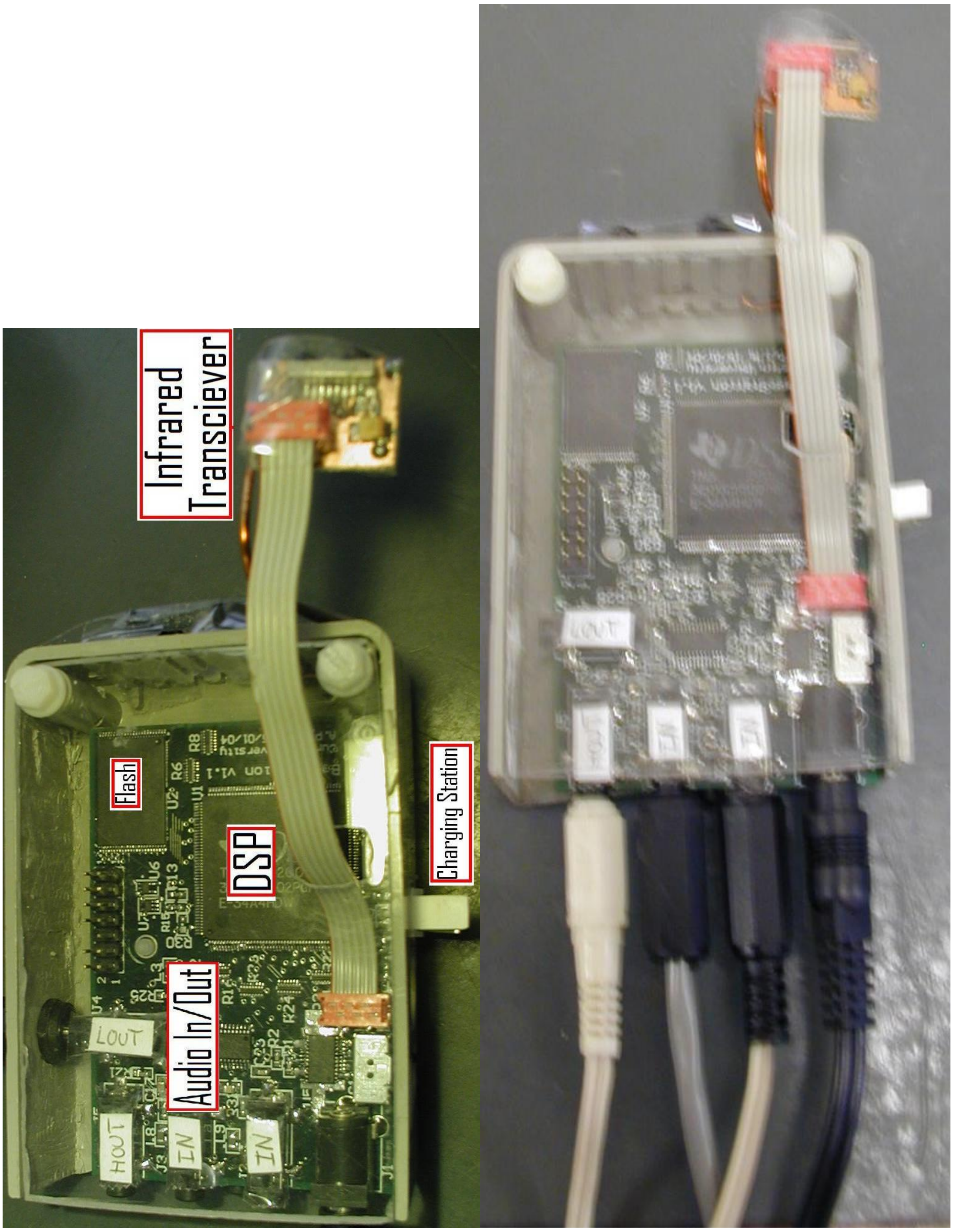


Figure 4-4: Base Station Prototype Layout

4.3.3 Headset prototype #1: headset mounted transceiver system

The mounting of the infrared transceivers onto the speaker earpieces will allow all the system components to be connected to the headset itself without the need for any cables, thus producing a neat and ergonomic unit. Figure 4-5 and appendix III.e show the basic hardware architecture and PCB circuitry of the headset system respectively.

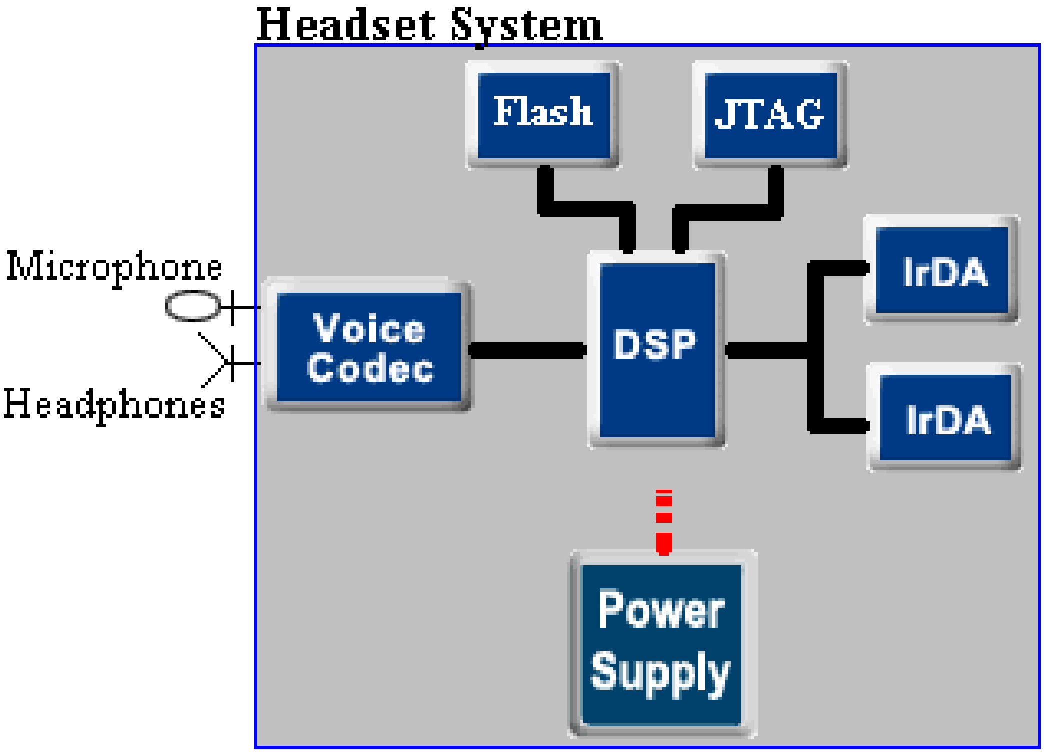


Figure 4-5: Headset Hardware Overview

This design has two IR transceivers either side of the head and hence implements a very basic form of the SDT technique. This provides link redundancy and helps reduce potential shadowing problems if the user rotates their head in the horizontal plane (yaw movement) or from blockage from another part of the body (a hand for example). Also, the exact angular alignment of the transmitters can be optimised via the stiff wire mount to suit the wearer’s requirements and in some way be employed to increase the horizontal angular field of view. See Figure 4-6, Figure 4-7, Figure 4-8 and Figure 4-9 below for actual test bed layout and profiles in use.

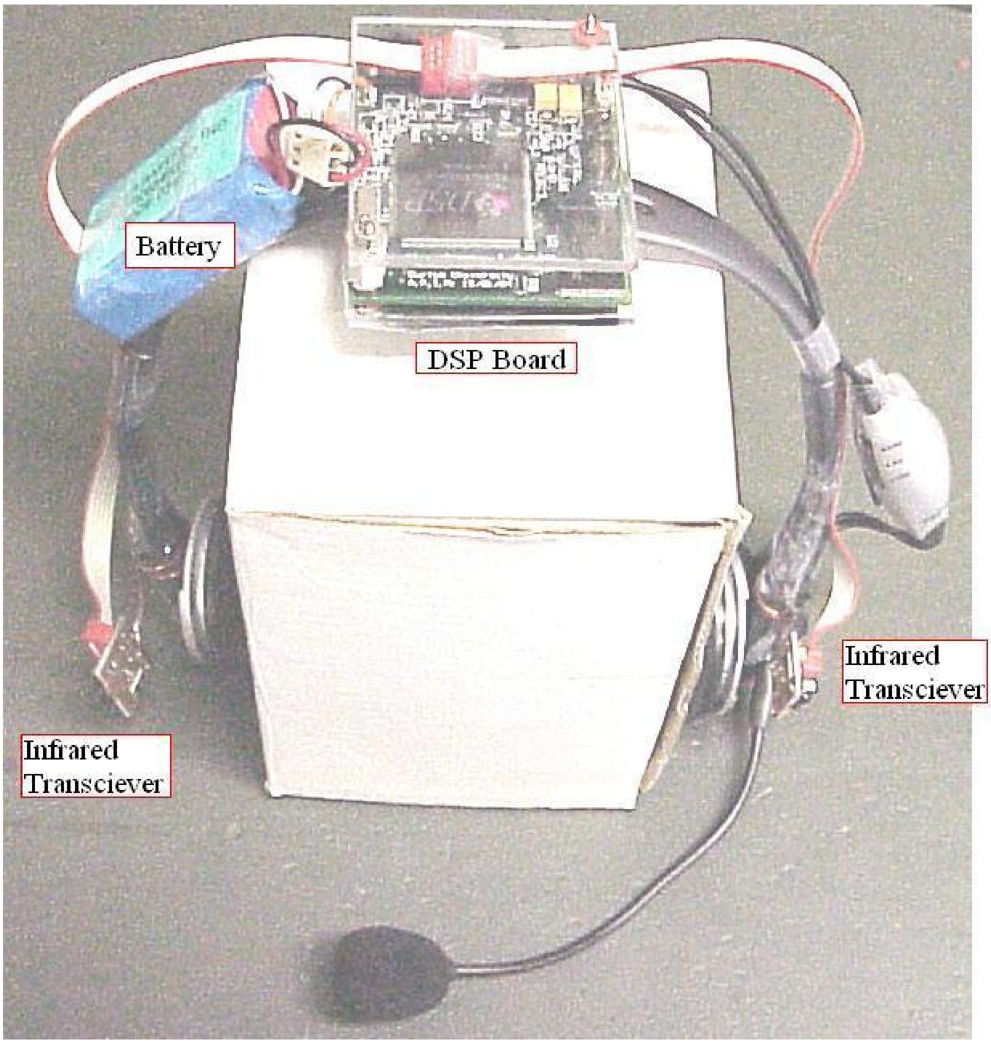


Figure 4-6: Headset Prototype #1 Layout

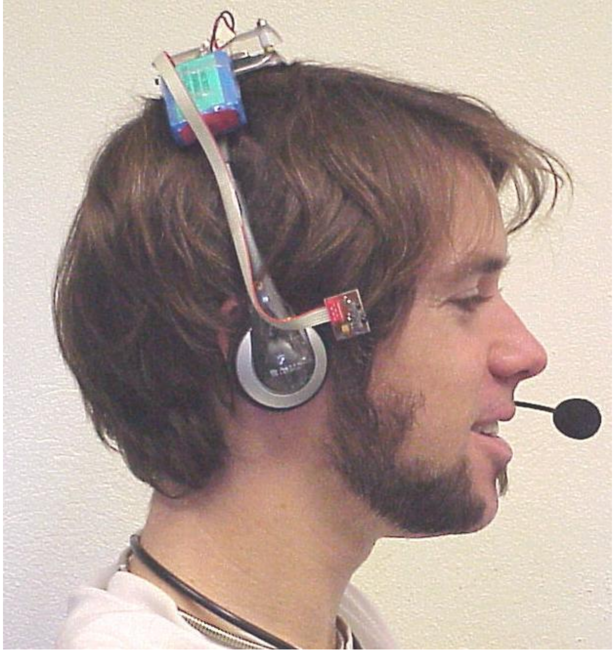


Figure 4-7: Headset Prototype #1 profile view

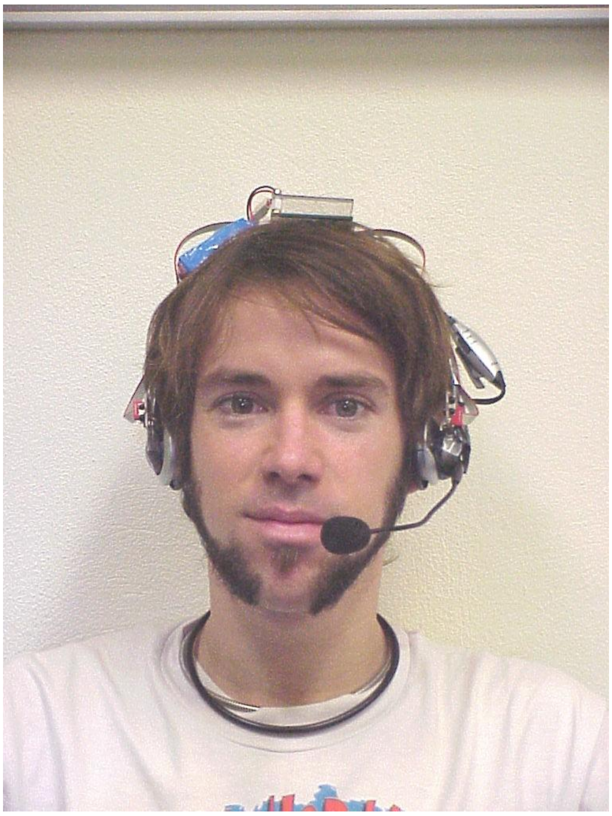


Figure 4-8: Headset Prototype #1 frontal view

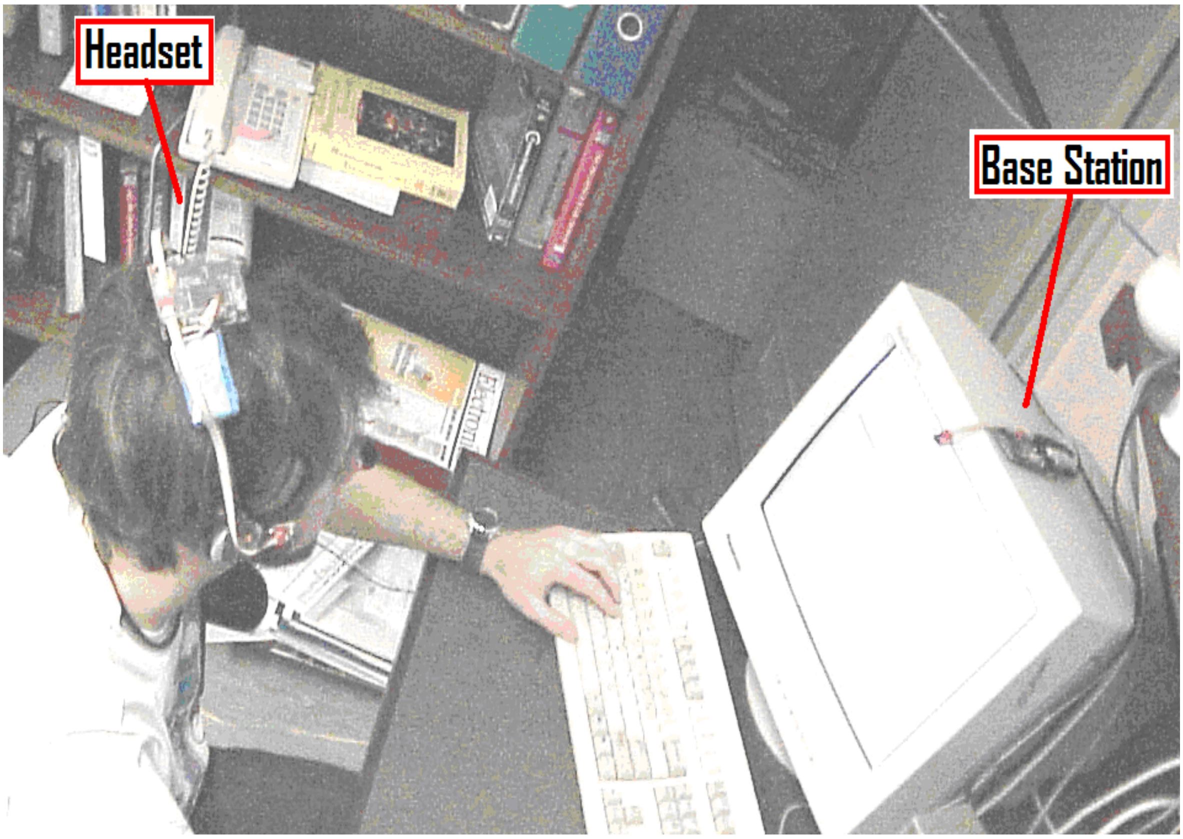


Figure 4-9: Headset Prototype #1 in use.

4.3.4 Headset prototype #2: pocket mounted transceiver and systems

The alternative mounting of the control hardware and transceiver within a shirt pocket allows the wearer to turn their head without loss of communication, and communication is only lost by angular movements of the body. This allows greater freedom for head gestures in conversation without loss of communication. Additionally, if the base and headset are correctly aligned, then the wearer's upper torso should act as a signal blocker and aid the implementation of a cellular network within the call centre environment. This design however, to its detriment, relies on the use of wires between shirt pocket and headset (raising possible tangling issues) and forces the user to wear clothing with a suitably located pocket. See Figure 4-10, Figure 4-11 and Figure 4-12 below for the prototype #2 layout and profiles in use.

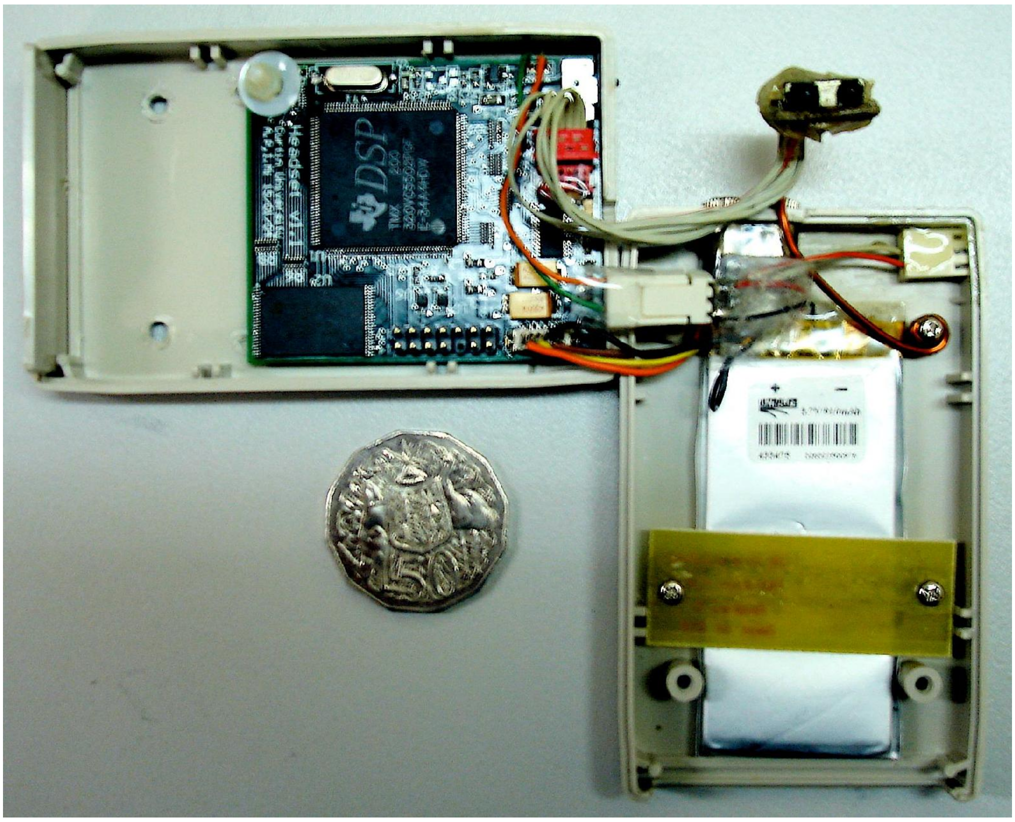


Figure 4-10: Headset Prototype #2 Layout

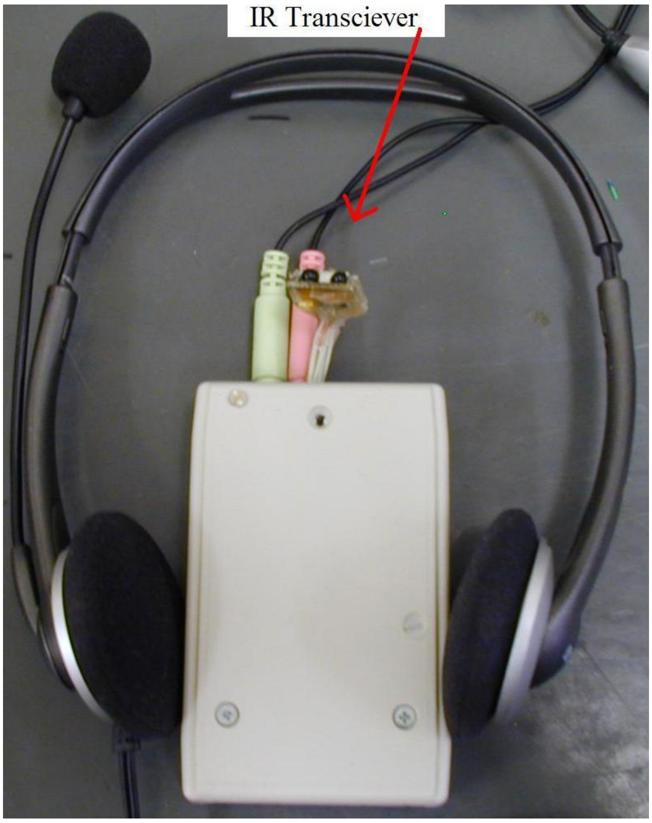


Figure 4-11: Headset Prototype #2 enclosed.

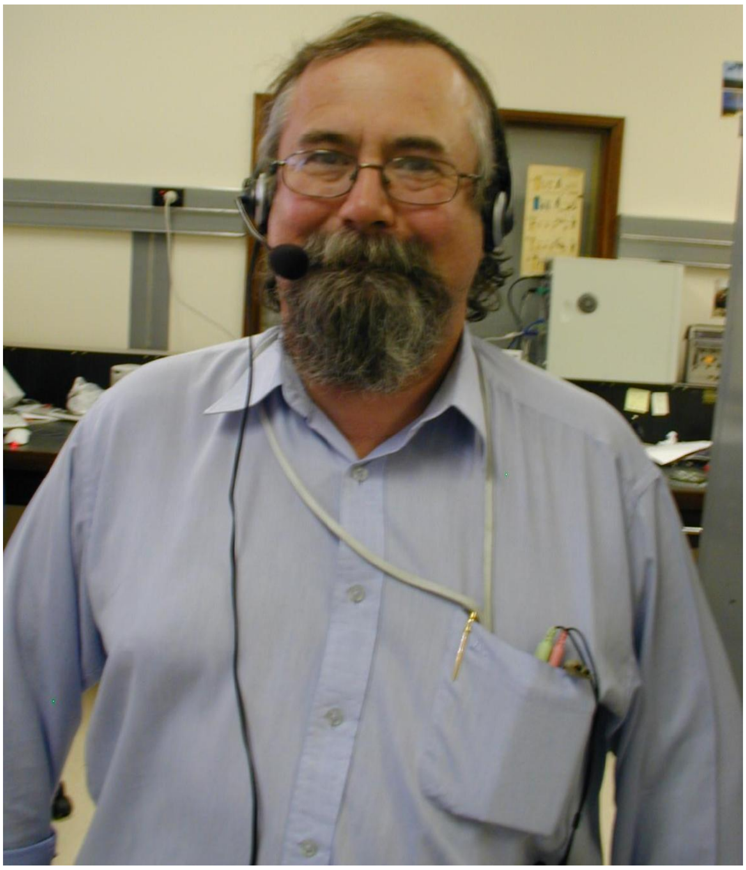


Figure 4-12: Headset Prototype #2 in use.

4.3.5 Prototype software: infrared wireless voice communications protocol

The software for the DSP controller was developed in the C programming language and the Texas Instruments Code Composer Studio Integrated Development Environment (IDE) was used. Appendix II lists the test bed base and headset source code. The software developed is a custom, voice centric communications protocol tailored to the requirements of VIPs. The software is optimised to reduce delay when using commercial infrared transceiver equipment which enforces half duplex communications with a minimum ‘send to receive’ turnaround time.

A flow chart of the controlling algorithm is depicted in Appendix I, and Figure 4-13 illustrates the periodic timing of the half duplex infrared channel.

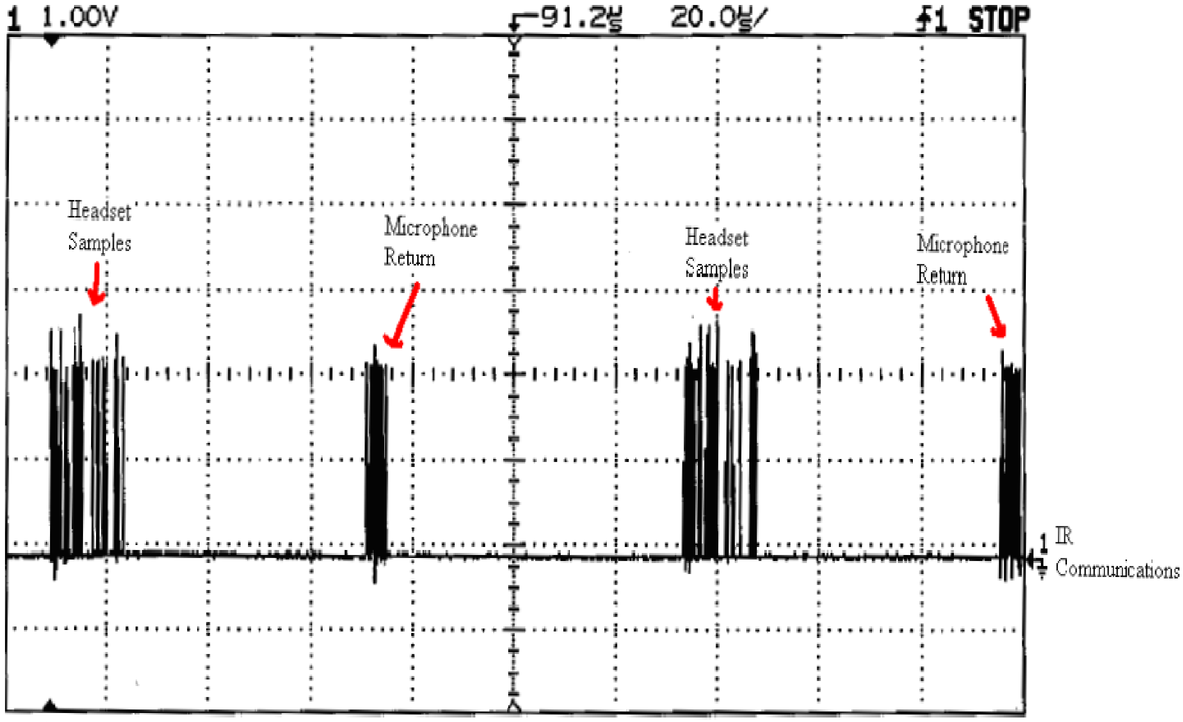


Figure 4-13: Infrared Channel half duplex timing

It may have been simpler and more flexible to utilise the IrDA protocol stack for the control algorithm. However the IrDA standard is tailored to the requirements of robust, multi-product data transfer, not real time audio. Its implementation would add significant processing and memory overhead resulting in extra processing delay and power consumption. The custom protocol does however exhibit very similar characteristics to the IrDA physical layer and also has the following features:

- Master-slave communications with the base station as the controller. This minimises headset power consumption and extends battery life by reducing processing and signal transmissions.
- Simple power control in which the headset does not transmit unless in communication with the paired base station (determined by a base ID code within the protocol header frame). This reduces spurious IR transmissions and inter-user interference if the user is not facing their computer terminal.
- CRC (cyclic redundancy check) error detection to prevent excessive bit and frame errors from producing uncomfortable levels of noise in the communicated audio. The protocol does however allow small bit errors within the least significant byte (LSB) of audio packets to pass into the audio stream. These bit errors largely result from poor signal to noise ratios from misalignment of the transmitting and receiving transceivers and the volume of noise increases as the angular misalignments become worse. This noise can be used as feedback to the wearer to re-align their position before communications are totally lost.

Due to time constraints, the particular code within Appendix II only supports an 8 KHz sampling rate and extra coding for FIFO software buffering is required to provide 16 KHz operation with the half duplex link. This however was deemed to have minimal impact on the ability to test the ergonomics and cellularity of the system. The actual raw bit rate of the developed system is 700kbps (which can easily be increased to 4Mbps) and once suitably coded, the bandwidth available is undoubtedly capable of providing the required audio quality.

4.4 Testing and verification findings

4.4.1 Effectiveness and ergonomics of IR in providing base-load wireless communications

4.4.1.1 Test objectives and setup

The effectiveness and ergonomics of infrared in providing base-load wireless headset communications was to be thoroughly tested on the equipment bench, within a simulated laboratory environment and then in actual use within a working call centre. However, due to time and prototype volume constraints, the headset was not able to be fully tested within the working call centre and trials were only conducted with a single VIP user.

All testing was performed with transceivers at full transmission power and the audio media included constant, single frequency sinusoidal audio (predominantly at 2 KHz) which allowed easy detection of bit and frame errors and repetitive synthesized voice for practical simulation. The test bed’s use of PCM audio was less likely to produce noticeable noise during the testing regime than a highly compressed audio format. The testing of a compressed format was not considered necessary however as the ample bandwidth capacity provided by the IrDA link and the greater need to maximise communications robustness and ergonomic freedom in transceiver alignment largely mitigated the usefulness of audio compression.

4.4.1.2 Bench test of IR transceivers

The bench tests and simulation tests, depicted in Figure 4-14 and Figure 4-9 respectively, were used to assess the physical layer attributes of the headset communications and the consequences transceiver misalignment (θ_B, θ_H) and distance (d) produce on the wireless bit stream (refer to Figure 4-15). The paradigm for the testing regime was that if the test scenario allowed even slight noise to be repeatedly subjectively detected in the audio stream then given the tight requirements for noise-free audio, the headset was considered to have failed under the link conditions.

The exact BER for various θ_B, θ_H and d was not measured with the prototype test bed due to difficulties in data recording with the DSP software IDE. This information may have proved to be useful for future design and implementation but was not considered a major detriment to the testing regime as actual human observation is the defining indication of link quality (Hammer 2003). It is also true however that this form of subjective testing may incur greater variance and inconsistency in results than exact BER plots, and in light of this, great care was taken to create an acoustically noise-free lab environment. For each of the test scenarios, this allowed much easier detection of link noise than would be possible in an acoustically noisy call centre and provided confidence that the lab results will sensibly translate to the intended environment.

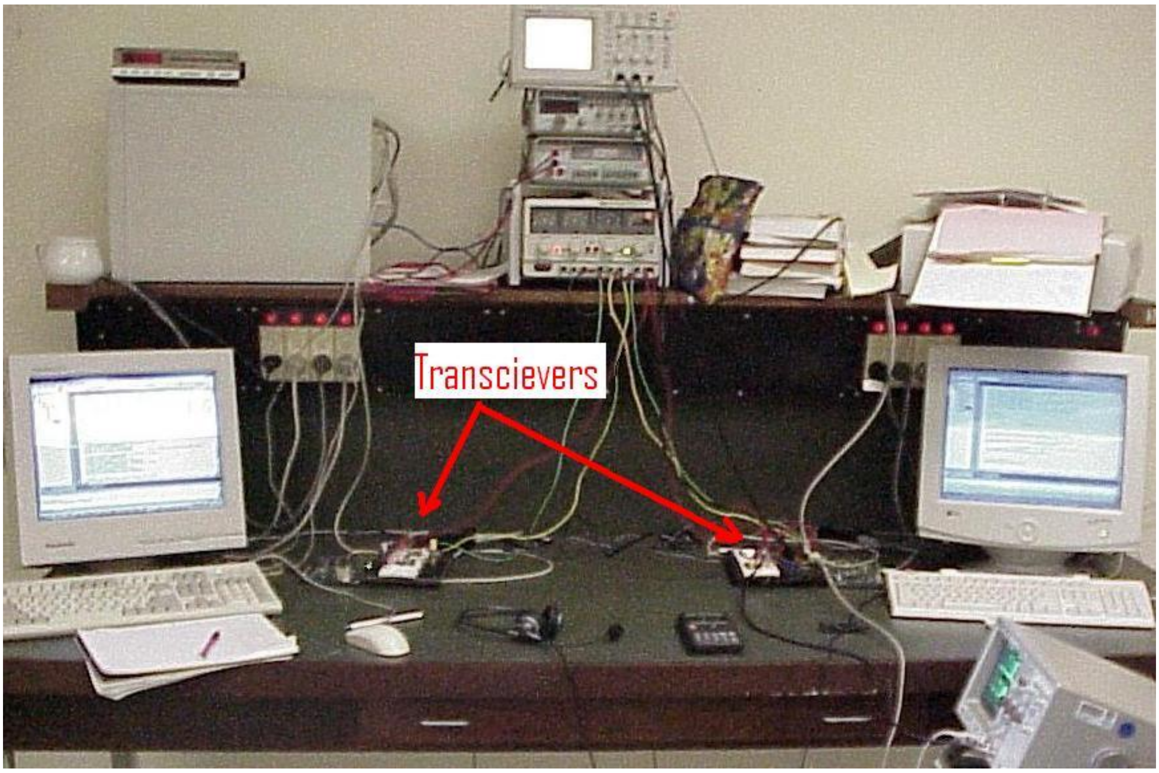


Figure 4-14: Test Bench Layout

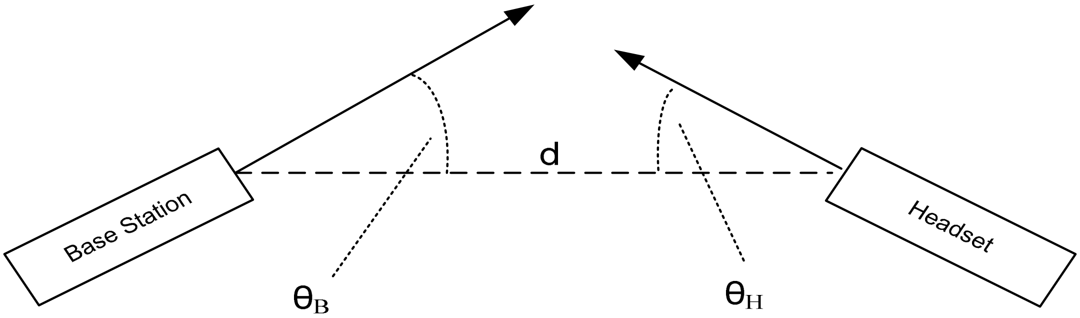


Figure 4-15: Base and Headset Link Geometry

From the bench tests, it was observed that for the likely scenario where $\theta_B = 0°$ and d = 1 meter, the communications link was robust and voice quality excellent when θ_H was less than 30°. Increasing incidence angle above this starts to affect the BER and create noticeable distortion in the audio. The 30 degree half angle is roughly double the IrDA standard of 15° and was most likely possible due to a combination of the low IR noise test environment, sensitive adaptive threshold controllers within the transceivers and the lack of an appropriate transceiver enclosure to limit the Tx and Rx field of view. The IrDA standard does specify a field limiting enclosure, see Appendix IV for a specific example, however it is evident some advantageous free play does exist with IrDA compliant transceivers to allow the exact field-of-view (FoV) to be optimized via an appropriate enclosure to suit the needs of the wireless headset.

Increasing the distance between the headset and base station was found to widen the lateral position window to some degree, but as depicted in Figure 3-17, this also brings negative impacts to the SNR and gradually introduces noise into the audio signal. This was also likely to be less of an issue for the low IR noise laboratory environment and the exact distance and hence lateral freedom will probably be much less within noisy call centre environments.

4.4.1.3 Practical test of effectiveness and ergonomics of prototype #1

Headset prototype #1 was designed with the intention that by mounting the transceivers on the earpieces and all the required components on the headset itself, the headset would be one contained unit with no external wiring. During simulated laboratory tests (refer to Figure 4-9) and brief exposure to the call centre environment, it was observed that the correct angular alignment of the transceivers was easy to setup for individual user requirements via appropriate bending of the stiff wire mounts. To allow the optimal freedom of movement within the Tx and Rx angular field of view, the best results were found to be when the base station was located at the same height of the IR transceivers on the headset and directly in front of the user.

Roll movements did not affect the unpolarised infrared link with the base station placed in front of the user and optimising the dual transceivers exact angular orientation allowed the horizontal FoV to be almost doubled to 60°. The dual transceivers also provided some link redundancy from head or hand movements blocking the LoS of one transceiver. Long, unrestrained hair did however raise blocking issues, often necessitating loose hair to be tied or clipped back.

Following initial user familiarity and training, the effort to maintain the correct head lateral position, pitch, and yaw for constant communication with prototype #1 is not excessive for short term use. However it is obvious that noise feedback and loss of communication from angular misalignments forces the user to maintain certain head orientations and long term (all day) use does raise potential ergonomic problems from the adoption of an unnatural posture. Furthermore, as VIPs are prompted of an incoming call by their headset, it would be impractical to assume that they would maintain the correct position all day. Hence it is apparent that an infrared only headset in the design of prototype #1 would be considered too unergonomic for VIPs working in call centre environments. Combined in an IR/RF hybrid design however, which does allow incidental human movements without significant consequence; it may act quite appropriately in carrying the base-load of communications.

4.4.1.4 Practical test of effectiveness and ergonomics of prototype #2

Prototype #2, which locates the control box and transceiver in the wearer’s pocket, was produced to allow natural head movements and envisaged to only suffer loss of communication from large and infrequent upper torso movements. Once the transceiver was aligned and the user seated, this design indeed made it possible to maintain a connection regardless of head movements, greatly improving ergonomics for all day use. However incidental misalignments were found to be caused by the following:

- Arm movements, especially if the pocket was located on the same side of the moving arm. This is of greater consequence however for sighted persons switching between mouse and keyboard than VIPs who primarily use a keyboard only.
- Transceiver movements within the pocket, especially if the control box is loosely fitted within the pocket.
- Rocking and swinging on a pivoted chair.

The fact that prototype #2 relies on the user wearing a shirt with a suitably located and sized pocket to allow frontal communication may raise issues for workplaces without a correctly designed uniform. Also the use of the headset to control box wires, whilst short and encompassed with the user, may introduce the tangling and breakage issues the wireless headset design was trying to avoid. These issues and the decreased effectiveness in providing base-load communications from the incidental movements listed above, suggest that prototype #2 may not be a suitable design for a hybrid IR/RF system.

4.4.2 Practical application and robustness of cellular IR communications in call centre environments

4.4.2.1 Test objectives and setup

The effect that multipath interference, headset design, geometrical layout and the use of partitions have on the ability to implement an IR cellular network in a call centre environment were tested by simulation on the equipment bench and in the laboratory environment. Brief tests were also performed in a working call centre. However due to time and prototype volume constraints the effects of inter-user interference from multiple headsets was not able to be tested.

Subjective testing of the scenario illustrated in Figure 4-16 was performed with varying user and interferer angles of incidence ($\theta_{B,}$ θ_I, θ_{IB} and θ_H) and separation distance for the user (d) and interferer (r). Headsets were not considered to be likely interferers, as any headset not in communication with its paired base was programmed to stop transmission and the probability of User 1 headset to User 2 base interference is unlikely with a 30 degree transmission angle, typical user densities and the use of office partitions. The paradigm for the testing regime was that if the test scenario allowed even slight noise to be repeatedly detected in the audio stream then given the tight user requirements for noise free audio, the cellularity of the infrared network was considered duly compromised.

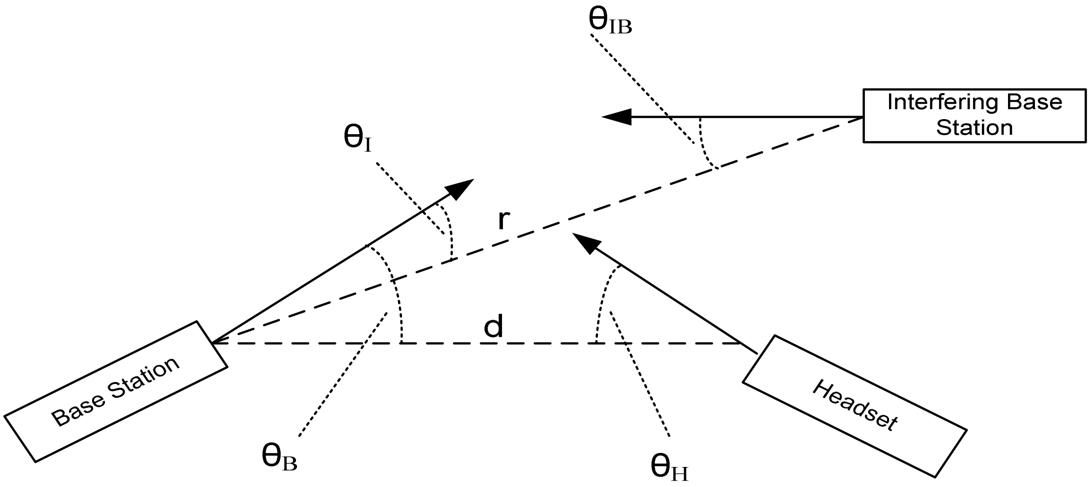


Figure 4-16: Base, Headset and Interferer Geometry

4.4.2.2 Tests results of the effect of multipath interference has on IR cellularity

Multipath interference from signal reflections was tested and deemed not to be a significant issue for common office partition materials such as carpeted, clothed or painted (grey) surfaces. In these tests, close range interference from reflected infrared was not able to be detected as audible noise on a direct LoS link, signifying the attenuation from reflection was great enough to not affect the BER. Some effect on the SNR must occur however and one test revealed how the reflectivity of human skin may add to multipath interference as it was possible to establish a reliable audio link by reflecting the transmitted IR off the palm of a hand onto a hidden receiver. This was most probably aided by the low noise environment and ATC of the infrared transceivers. For commercial deployments, a precision test apparatus (not developed) capable of measuring infrared signal strength will greatly aid implementation by ensuring other untested partition materials that may be used do not have reflection coefficients that may create a noisy environment.

4.4.2.3 Tests results of the effect headset design has on IR cellularity

The design of headset prototype #1 required the base station to be setup at the user’s head height (for example on top of a PC tower or monitor) to maximise the lateral freedom of movement. However this also leads to minimal blocking of the signal by the wearer’s body and can allow opposing base stations to have direct LoS with each other, thereby degrading the SNR. Also if the partitions walls are lower than a user’s seated head height then LoS may be possible between headsets of users facing each other on either side of a partition wall.

The use of dual transceivers on headset prototype #1 highlighted the need for the DSP to perform independent CRC analysis and base ID code checks for each of the transceivers received data. Noise from reflections and other sources (sunlight and fluorescent lighting for example) can easily be picked up and transferred to the audio stream by the ATC circuitry of a transceiver not in LoS with the base. This independent functionality was not performed with prototype #1 and its implementation will greatly improve the audio quality in these circumstances.

The design of Prototype #2 requires the base station to be setup at the user's chest height to maximise the lateral freedom of movement. This greatly aids the blocking of the base transmitted signal by the wearer's body and reduces the probability of direct LoS for opposing base stations. If however one user moves from their desk, the constant base "wake up headset" transmissions required for the master slave protocol will introduce interference to the opposing base station.

4.4.2.4 Tests results of the effect of geometrical layout and partitions have on IR cellularity

Typical real life call centre layouts depicted by Figure 3-18 and Figure 3-19 are usually characterised by a base station to headset distance (d) of 1m. In the scenario where $\theta_B = \theta_H = \theta_I = \theta_{IB} = 0°$, it was found that noise within sinusoidal audio was subjectively detectable when interferer distance 'r' was less than 3.5 meters. The noise detected was most probably due to bit errors in the most significant byte (MSB) of audio data or frame errors from corruption of the protocol header. This 3.5m separation is more forgiving than the theoretical interference polar plot illustrated in Figure 3-17 and confirms that uncompressed PCM audio can allow much higher BERs than 10^{-7} and still provide quality audio. Also, certain techniques (not implemented) such as forward error correction (for header frames and MSBs) and an advanced power control where transceiver transmission power is adjusted with decreasing BER may allow further reduction in the base to base distance.

In practice human yaw and pitch movements will cause θ_H to be something normally greater than 0°. In the worst case scenario where θ_I and θ_{IB} remain at zero degrees, it was found that when $\theta_H = 15°$, which nominally represents a minimum angular free play to maintain ergonomic headset functionality, noise was able to subjectively detected in the audio stream when r was less than roughly 5 meters. This worst case scenario must be catered for and it is likely that call centres layouts depicted by Figure 3-18 and Figure 3-19 with interfering base stations less than 5 meters apart will not have reliable infrared communications. In respect to an IR/RF hybrid design, this will cause frequent handover to the backup RF PAN and reduce audio quality.

The observations and tests highlight the compromises and impacts for existing standard call centre layouts are;

- if a headset design is based on Prototype #1, partition walls must be higher than the users seated head height.
- in low density call centre environments;
    - for layouts depicted by Figure 3-18, opposing base stations need to be at least 5-6m apart. (Not considering FEC and advanced power control);
    - for layouts depicted by Figure 3-19 where the side partitions do not block User 1 to User 5 base station LOS, then;
        - Diagonally opposing base stations need to be at least 5-6m apart. (Not considering FEC and advanced power control);

        - The partitions must be at least long enough to prevent LoS between User 2 headset and User 1 base station.

- in high density call centre environments, where there is less than 5-6m base station separation then;
    - the layout in Figure 3-18 will not likely be able to provide the required cellularity for link robustness and a multiple access scheme will be necessary (TDMA, CSMA/CD or CDMA for example). The effects that the multiple access scheme has to bandwidth, power requirements and complexity will also need to be considered and it is possible that IR may not provide a viable solution;
    - the layout in Figure 3-19 must utilise side partitions that are long enough to block User 1 base-station to User 5 base-station LoS for the given user seating/desk angles.

If it is possible to design a new call centre or perform modification to an existing one, then there are several options which can be pursued to maximise the cellularity of the infrared network. Ideally the layout would involve fully enclosed cubicles where there is a partition wall behind the user and consequently there would be little interference to affect the robustness of the infrared communications. Some multipath interference may still exist from ceiling reflections, but as tested, this is generally not considered great enough to affect a direct LoS link. This layout will require greater area per employee to allow suitably sized cubicles and walkways (relative to open designs), thus limiting user densities and possibly incurring additional costs to a call centre.

If user density and costs are a considerable factor in the design of a call centre, then the implementation of the layout in Figure 4-17 below may bring significant benefits. The layout exploits the angle diversity plotted in Figure 3-17 to ensure base-station:base-station, base-station:headset and headset:headset interference is unlikely.

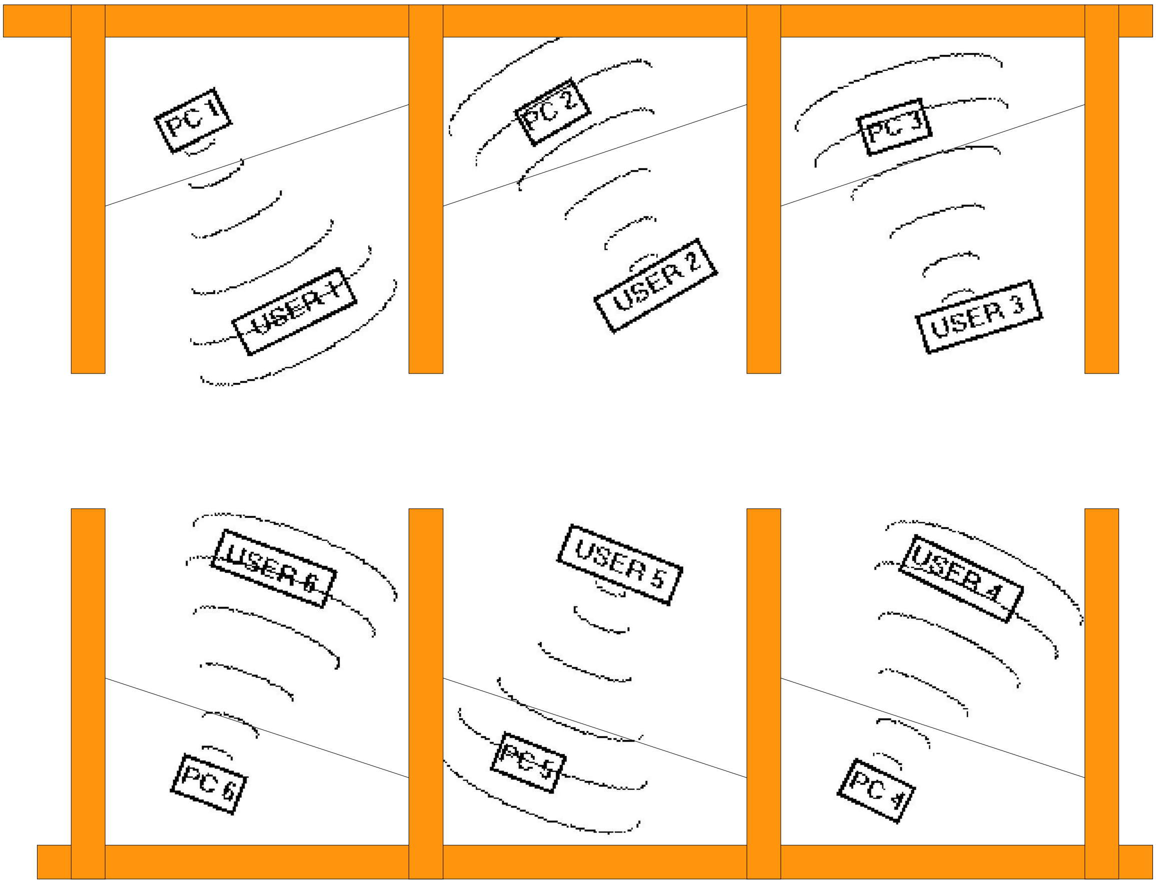


Figure 4-17: Optimal user layout for IR cellular architecture

Tests were undertaken to deduce the level of interference for the scenario where d = 1m, $\theta_B = 0$, and $\theta_I = \theta_{IB} = 35°$, thus simulating a realistic seating/desk angle of 35°. The 35° angle was chosen as it lies outside of the IrDA minimum transmission half angle of 30°, and if a suitable transceiver enclosure is utilised, then User 1 to User 6 base stations should theoretically not be able to have direct LOS.

In tests of the worst case scenario where $\theta_H = 15°$ (where a user has yawed or pitched) noise was not able to be detected in the audio stream until the interferer distance r was less than two meters. A base station to base station distance of two meters would require the headset users to be back to back, an unlikely scenario and thus it can be safely assumed that the layout can allow robust communications in very high density call centre environments. Also, the fact that noise was able to be detected at all was most probably due to the lack of an appropriate transceiver enclosure to limit the Tx and Rx field to 30° and not SNR degradation from multipath interference. Once a suitable enclosure is implemented (see Appendix IV for details of appropriate enclosure design) the base station to base station distance (r) should be negligible for a seating angle of 35° and even greater headset angular freedom without noise should be possible.

The layout does not logically appear to have any characteristics that would make it any more difficult to setup or use compared to other standard call centre arrangements, and its geometry presents a means by which an infrared link can reliably and ergonomically provide base-load communications in an IR/RF hybrid design.

5.0 Conclusions

5.1 Introduction

The main objective of the research was to determine the optimum solution for meeting the requirements of a wireless headset for VIPs in multi-user environments. It determined that a headset combining directed LoS infrared for base-load communications with a RF PAN acting as backup provided a credible solution. Areas of concern were highlighted with the proposed system, most notably being whether IR could provide ergonomic and robust base-load headset communications within a call centre environment. A working, functional and flexible test bed was developed to assess and document the capabilities and compromises of the proposed design.

5.2 Contributions

The research undertaken in this thesis found that an IR-only headset will fail to meet reasonable ergonomic requirements on its own. Natural head and body movements cause loss of communication and alternatively, if the user attempts to meticulously maintain the correct position, then stiff neck and other ergonomic problems can arise. Further, the requirement that the VIP must maintain LoS at all times to be able to receive prompt of an incoming call is also quite impractical.

If the infrared communications are combined with a backup RF PAN as a IR/RF hybrid however, then for a user operating a computer terminal where the probability of LoS communication is high, the infrared medium will be quite suitable in providing the base load of communications and in turn allow;

- high bandwidth, interference free and low delay audio with exceptional quality;

- a backup RF PAN that is unlikely to experience congestion given the low percentage of time the RF PAN is operating and the low probability that adjacent users will both be in backup mode at the same time;
- highly secure communications in IR mode and RF backup mode with the use of buffered OTP keys.

Tests to ascertain how the required infrared cellular network could be implemented in a call centre environment and its robustness in application found that compromises will exist for standard call centre layouts particularly in the angle of users, the distance between them and the partition walls that separate them. Most pertinent, for high density, open and un-angled layouts where modification is not possible, multiple access techniques will need to be implemented. Considering the disadvantages these techniques bring, it is likely that a hybrid IR/RF headset will not provide the best solution for VIPs in such call centre environments. Conversely, for call centres with enclosed cubicles or angled seating, particularly with a layout depicted in Figure 4-17, infrared can quite easily and robustly provide base-load communications for the headset and in doing so present a suitable wireless headset design to meet the unique requirements of VIPs in multi-user environments.

When compared to alternatives such as NFMC and radio only communications, the IR/RF hybrid headset is more complicated and not as flexible in deployment. It is also quite disadvantaged in that it may not be suitable for un-angled, open call centres. However if angled layouts are used or if modification is possible to unsuitable layouts, then the IR/RF hybrid has major advantages in terms of audio quality, reliability and security in multistorey buildings and in this regard, it is the most capable in meeting the strict requirements of VIPs working in multi-user environments. Furthermore, when compared to NFMC it is not affected by intellectual property design and cost implications.

5.3 Recommendations for future development

5.3.1 IR / RF PAN hybrid development

To allow testing and total proof of concept of the IR/RF hybrid wireless headset design, the design and integration of the backup RF PAN is required. The use of the Bluetooth standard is quite suitable due to its low power, flexible and robust qualities. However further testing will still need to be performed to document its abilities in high density environments with random sources of broad spectrum interference (802.11 for example). There is also the requirement for the development of hardware and software to;

- provide fast, efficient and reliable handover between the IR and RF PAN, a non-trivial exercise, although Hou et al (2006) proposes a possible solution;
- handle the forwarding, buffering and use of infrared OTP keys to ensure security of communications when operating with the RF PAN;
- implement efficient and robust audio compression to reduce the bandwidth requirements and the probability of congestion when using the backup RF PAN plus G.711 standard VAD for similar benefits.

5.3.2 Improvements to infrared communications design and implementation

5.3.2.1 Development of precision test apparatus for infrared signal strength and link BER.

To allow more objective testing and precision design, there is a need to develop an accurate test apparatus that will allow recording of infrared signal strengths and also a wireless link's BER and PERs. Whilst VIP observation is the most valued source of critique for the overall operation of the headset, objective measures will allow a more exacting approach to understanding the influence of;

- enclosure design and window materials;
- interference from window light and EMI;
- reflection coefficients of office and partition materials.

5.3.2.2 Completion of physical and link layer software

The software protocol used in the test bed prototypes is far from complete and requires further development to be considered suitable for application in a call centre. Items necessitating further development include the following:

- The coding of large and flexible FIFO buffers to allow the provision of the required 16 KHz sampling rate, 16-bit audio.
- The design and coding of a flexible base-station to headset pairing protocol.
- The ability to perform independent CRC analysis and base ID code checks for each of the transceivers in headset prototype #1. This will prevent noise from reflections and other sources (sunlight and fluorescent

lighting for example) from being picked up by a non LoS transceiver and then being transferred to the audio stream.

- Advanced transmission power control: It is envisaged that for low bit error rates that transmission power can be reduced, thus saving battery life and limiting the level of interference to other users. An example method is described in page 34 of Agilent 2003b.
- Forward error correction: Due to the real time audio requirements and sensitivity to link delay, it is unlikely that resending of corrupted wireless packets will be possible. Forward error correction, whereby redundancy for control bits and the most significant bytes of data in the audio is implemented in the packet stream will allow higher quality audio for a given BER.
- Noise mitigation: During testing, it was observed that the noise from interference or from losing LoS is particularly abrupt and intrusive. Due to the importance of preventing acoustic shock syndrome within the call centre environment, a fast and seamless handover to the RF PAN in an IR/RF hybrid design must be implemented. If this is not achievable then white noise, jump ahead or waveform fill techniques will need to be utilised in response to packet errors. Zero stuffing was tested and deemed not suitable as it generated high levels of noise in waveforms composed of randomly healthy and unhealthy packets (which are zeroed).

5.3.2.3 Sampling slip compensation for base station and headset DSP clock frequency misalignment

The crystals used to provide the DSP clock frequency for the test bed are set to supply a stable 12.0000MHz source. However due to temperature variations the exact crystal clock can vary by ±50 pulses per million and the slight variations between base station and headset crystals will cause the audio sampling frequencies to be slightly different. This results in frequently missed audio samples for the unit with the slightly faster clock frequency and noticeable noise from large phase errors as depicted in Figure 5-1

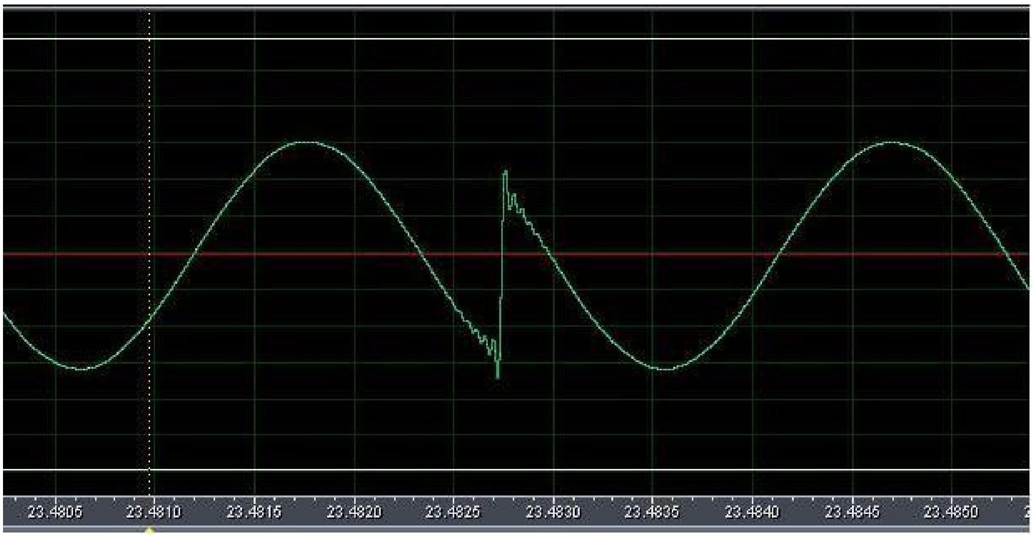


Figure 5-1: Waveform Noise with Sampling Slip

It was attempted during development to reduce the effect of clock slip in the headset test bed by the use of oversampling and decimation filters which can spread the clock mismatch over a greater number of samples. For example;

- A 4 KHz waveform is sampled by a base-station analogue-to-digital-converter (ADC) running at 8MHz. Upon receiving an IR transmission positive edge, a sum of samples in the codec buffer is then passed through an 8 KHz decimation filter.
- The filtered samples are then wirelessly transmitted to the headset.
- An interpolating filter then converts the 8KHz signal back to 8MHz for the digital-to-analogue-converter (DAC).

Hence, if skipping or repeating samples occurs due to clock slip the greatest phase error will be 0.18 degrees (4000 / 8000,000 *360). This level of phase error would be near undetectable to the user when compared to uniform ADC/DAC sampling at 8 KHz.

Due to clock frequency limitations with the hardware codec used, an oversampling frequency of only 96 KHz was possible. In this case, the maximum phase error is 4000 / 96000 *360, or 15 degrees and this is illustrated in Figure 5-2.

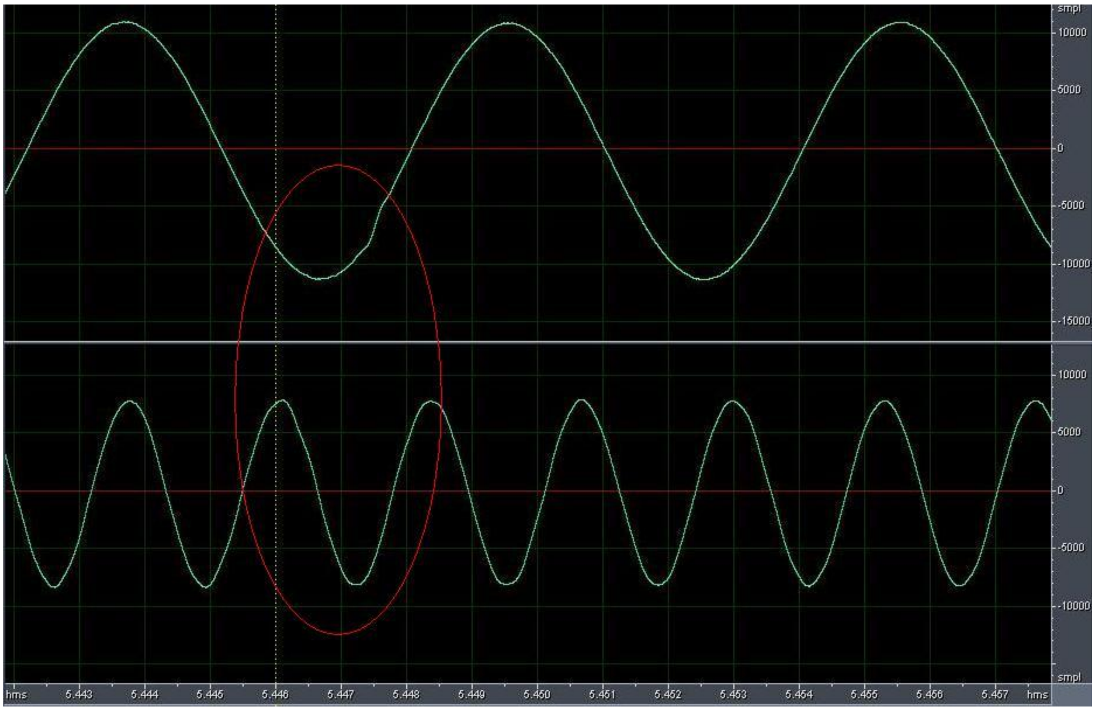


Figure 5-2: Waveform noise with oversampling and decimation filtering

While not of significant amplitude, this phase error is perceptible to the human ear and as such more advanced techniques to compensate for clock slip need to be implemented. A good solution is a Digital Phase Locked loop (DPLL) which can adjust the phase of the system clock with the rising edge of infrared transmissions, as illustrated in Figure 5-3 and Figure 5-4 below.

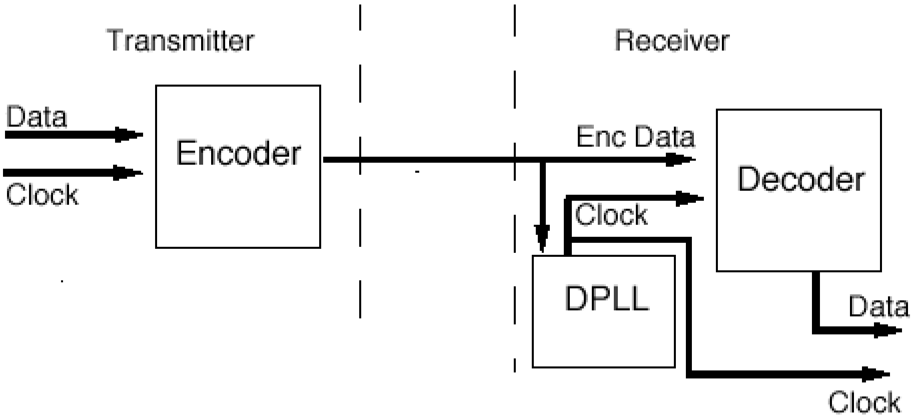


Figure 5-3: DPLL Hardware (FairHurst, G 2001)

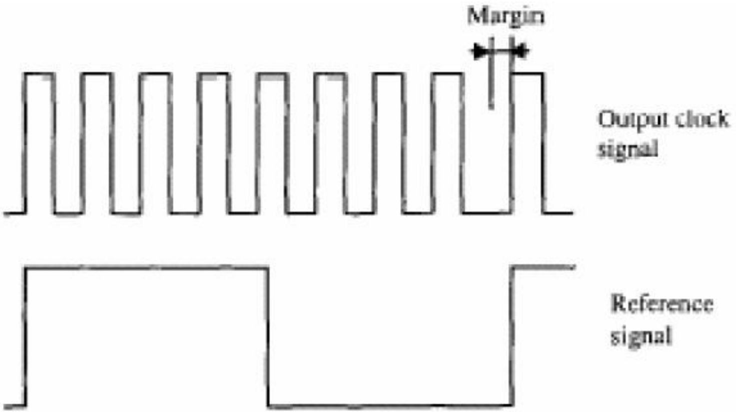


Figure 5-4: DPLL compensation of clock slip (Chew, M et al 2002)

5.3.2.4 Hardware considerations with Commercial of the Shelf Equipment (COTS)

5.3.2.4.1 IR transmission intensity effect on receiver turnaround time

During testing with headset prototype #1, it was found that sporadic clicking noises were apparent in the audio sent from the headset to the base whenever the link distance was less than 50cm. Sample captures revealed that the first bit in the protocol preamble was often being missed, see Figure 5-5, thus causing incorrect bit offsets in decoding. Without CRC elimination, the effect on the sinusoidal waveform is seen in Figure 5-6.

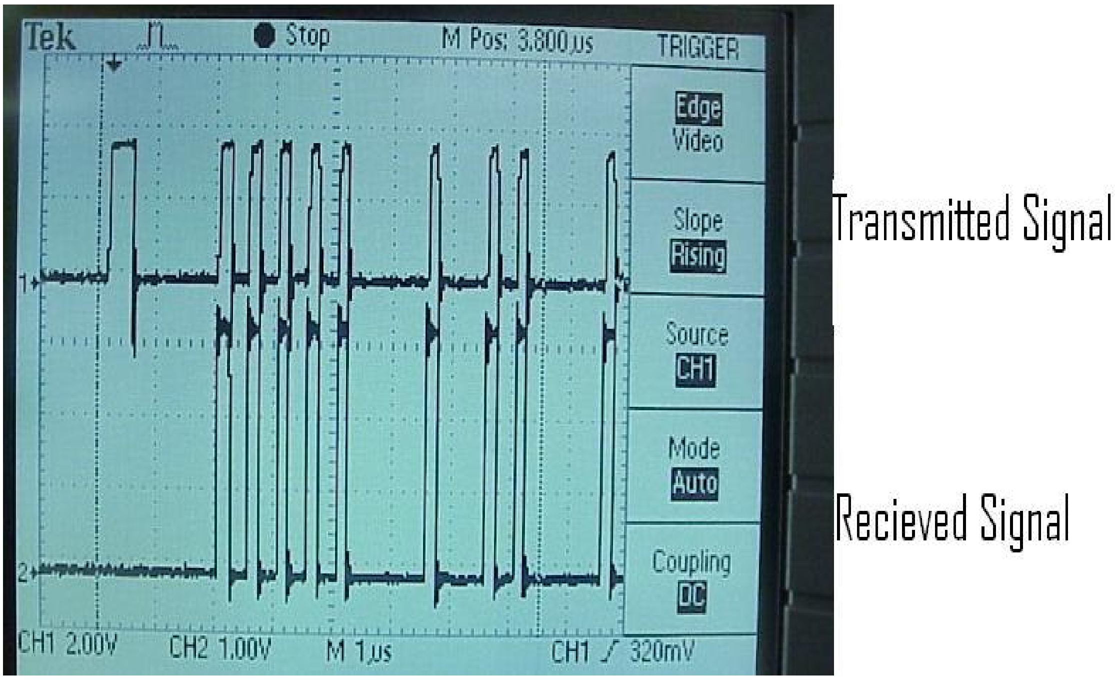


Figure 5-5: Receiver missing bit under high transmission power

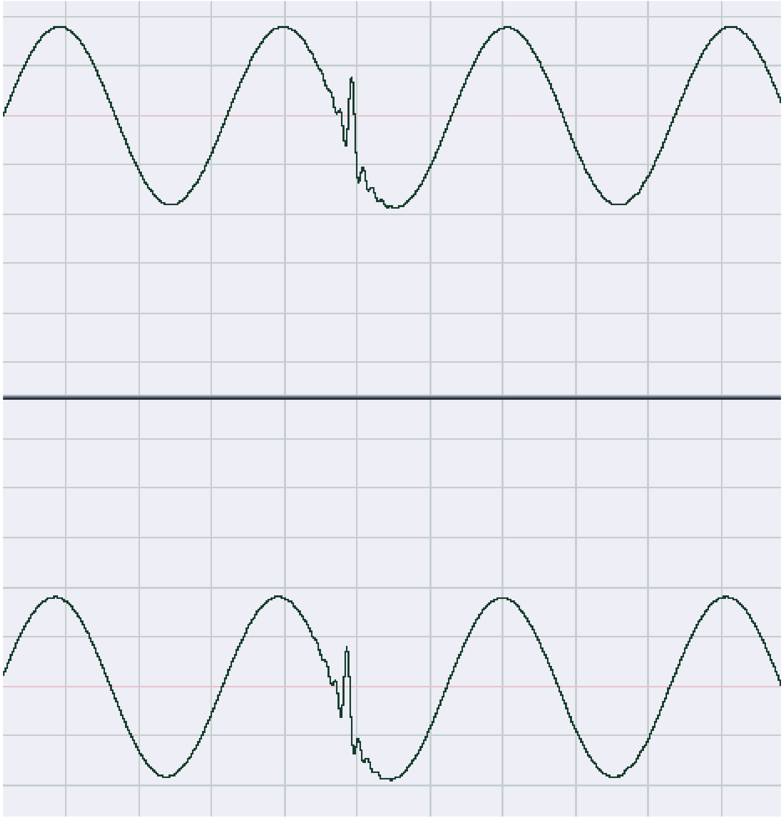


Figure 5-6: Receiver Overload 'Click' Noise Waveform

It was found that the noise could be prevented by reducing the power of the transceivers or by blocking one of the headset transceivers (by hand or rotation of the head). Even though this suggests a receiver overload situation, this was unlikely to be occurring as at full power each transmitter is only capable of producing 250 mW per steradian, or 500mW combined. At 50cm, this only equates to 0.2mW/cm^2 (refer to page 65 of Agilent 2003b), and Agilent 2003a rates the receiver maximum intensity at 500 mW/cm^2

It was also found that cutting the sample rate of the headset system by half to 4kHz, which then also halves the protocols half-duplex packet rate, prevented the noise even with full power, close range transmissions. It can be deduced from this that receiver turnaround sensitivity is the likely issue. Following LED transmission, where the transceiver's receiver is blinded by its own light, the ATC may be improperly adjusted. Agilent (2003a) lists this time as typically 40uS, and a maximum of 50uS. At the 8 KHz sample rate used, the turnaround time of the software protocol is 50uS and with high power transmissions, this may have allowed this constraint to surface.

This issue, then, can be easily avoided by either;

- increasing the software buffering (via a FIFO buffer) and reducing the turnaround rate of the half duplex link to be greater than 50uS (but consideration must be given however to ensure the total link delay remains below 150mS, as discussed in 2.2.3.4);

- utilising a protocol preamble consisting of a set of dummy bits followed by a unique code, where the DSP software should be able to identify the preamble code and add a bit offset if the first bit is missing or not.

5.3.2.4.2 Infrared transmission switching current effect on power supply noise

During testing, it was found that a small background noise was apparent in the headphones audio when the transceiver transmitters were set to full power (the power level is selectable by DIP switch). Waveform capture found that a slight glitch in the audio wave was occurring about every 0.0004s or 2500Hz. Refer to Figure 5-7 below.

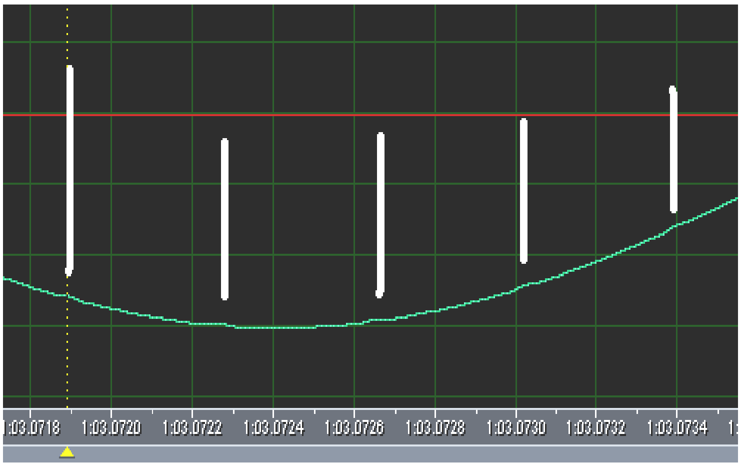


Figure 5-7: Headset Noise with transmitters at full power

Performing the same test with the headset transmitters turned off yielded a clean headphone signal, as seen below in Figure 5-8.

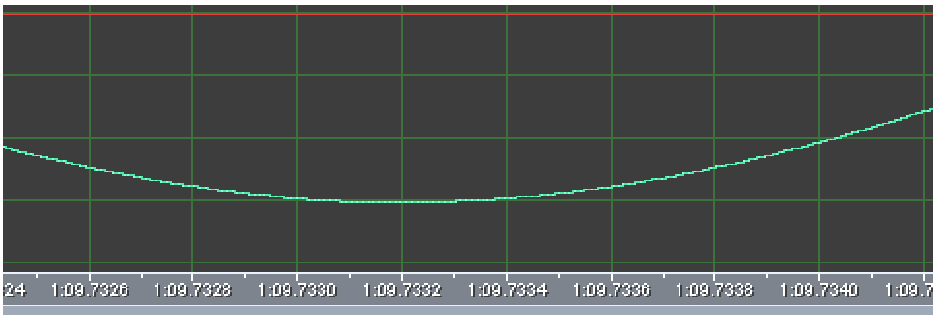


Figure 5-8: Headset waveform with transmitters off

Since the headset protocol sends 3 microphone samples at once, for an 8 KHz sample rate the headset transmitting frequency is 2666Hz or very close to the frequency of the waveform glitch frequency. It was obvious that the large LED switching currents of 650mA per transceiver, or 1.3A combined for headset prototype #1, were inducing noise into the power supply which is then transferred to the audio circuitry.

The problem was mitigated by digitally amplifying the headphone audio (by codec parameters) before analogue amplification. This requires great care in the signal encoding, transmission and decoding to ensure waveform cropping does not occur. For a commercial design, better techniques exist to prevent power supply noise such as;

- electronic separation of analogue and digital power supplies and, if possible, the use of larger rectifying capacitors;
- the use of G.711 standard Voice Activity Detection (VAD) to ensure the headset only transmits IR packets when the user is speaking so ensuring noise free audio while the user is only listening and so possibly also reducing power and bandwidth requirements when utilising the backup RF PAN in a hybrid design.;
- the use of advanced power control techniques;
- the transmission of IR only from the transceiver in LoS for headset prototype #1.

5.3.2.5 LOS improvements

The infrared transceivers used for testing in this project were designed for generic IrDA applications and are not optimised to suit the requirements of a headset. As such there are several methods which can be pursued to optimise the design of infrared transceivers for a wireless headset and these include;

- the use of concave lens in front of transceivers to create an elliptical FoV (Refer to Figure 5-9). Given that the ideal orientation for the base station is in front of the user and at the same height of the headset, the low frequency of human roll movements may allow the FoV to be vertically widened to give the user greater freedom in pitch angle. A vertically biased elliptical FoV is unlikely to affect cellularity in the call centre environment as LoS interference is improbable in the vertical plane. A small degree of extra multipath noise from the ceiling may result however, as well as a lower transceiver optical gain from the increased FoV;

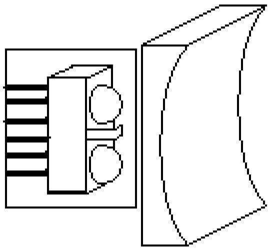


Figure 5-9: Transceiver Concave Lens

- if the intended environment is partitioned and standard power control is used then, given the low probability of headset to headset interference, it may be ergonomically advantageous to have a greater FoV for the headset compared to the base-station. While this may reduce the optical gain of the headset transceiver and decrease the SNR, significant ergonomic gains in headset angular freedom may be possible.

5.3.3 Noise Cancellation

For high density call centre environments with loud levels of acoustic noise, noise cancelling headsets can be of significant benefit to VIPs. The powerful DSPs used in the headset can be adapted to provide noise cancelling functionality and attenuate acoustic noise. The benefits include;

- increased intelligibility of high speed text to speech synthesized voice;
- lower required earpiece volume which will reduce ear fatigue and a wearer’s predisposition to acoustic shock syndrome.

6.0 References

ABCNews 2007, *Indian call centre*, Retrieved: December 2007, from http://www.abc.net.au/news/photos/2007/08/31/2020229.htm

Agilent 2003a, *Agilent HSDL-3602 IrDA® Data 1.4 Compliant 4 Mb/s 3V Infrared Transceiver Data Sheet*, Retrieved: June 2003, from www.agilent.com/semiconductors

Agilent 2003b, *Agilent IrDA Data Link Design Guide*, Retrieved: June 2003, from http://www.agilent.com/view/ir

Apple 2009, *VoiceOver in Depth*, Retrieved: April 2009, from http://www.apple.com/accessibility/voiceover/

Ashraf, I, Gkelias, A, Dohler M & Aghvami, A 2006, *Time-synchronised multi-piconet Bluetooth environments*, IEE Proceedings on Communications. Vol. 153, No. 3, June 2006

AudioLinkServices 2003, *Agilent Califone CLS-700L - wireless mic headset*, Retrieved: May 2003, from http://www.audiolinks.com/tek9/tek9.asp?pg=products&specific=joengne8

Auracomm 2003a, *Near-Field Magnetic Communication Properties*, Aura Communications Inc, Retrieved: November 2003, from http://auracomm.com

Auracomm 2003b, *LibertyLinkTM Summary Datasheet*, Aura Communications Inc, Retrieved: November 2003, from http://auracomm.com

Auracomm 2003c, *Nearfield magnetic communication for low-power wireless device,* Aura Communications Inc, Retrieved: November 2003, from http://www2.electronicproducts.com/Nearfield_magnetic_communication_for_low-power_wireless_devices-article-FEBAUR1-FEB2003.aspx

Barker, P, Boucouvalas, A 1999, *Effect of Random Alignment Sway on the Performance of IRDA Handheld Devices,* IEE Colloquium Optical Wireless Communications, Ref. No. 1999/128, Page(s): 9/1 - 9/4

Barker, P, Boucouvalas, A 2002, *Interference Induced Asymmetry in IrDA 4PPM Infrared Wireless Links,* Multimedia Communications Research Group, Bournemouth University, UK

BBC 2007, *Q&A: Wi-fi health concerns,* Retrieved: October 2007, from http://news.bbc.co.uk/1/hi/technology/6677051.stm

Bluetooth 2007, *Archtecture Radio,* Retrieved: June 2007, from http://www.bluetooth.com/Bluetooth/Technology/Works/Architecture__Radio.htm

Castillo-Vazquez, M & Puerta-Notario, A 2003, *Self-orienting receiver for indoor wireless infrared links at high bit rates,* The 57th IEEE Semiannual on

Vehicular Technology Conference, Publication Date: 22-25 April 2003, Volume: 3, On page(s): 1600- 1604

Chew, M & Pletcher, N 2002, *A Comparison of Various Clock Synchronization Techniques for Wireless Sensor Networks*, EECS Group, Berkeley University, USA

Cisco 2006, VoIP: An In-Depth Analysis, Retrieved: April 2009, from http://www.ciscopress.com/articles/article.asp?p=606583

Cojocaru, C 2005, *Low Power Blutooth for Headset Applications*, IEEE 9th International Database Engineering & Application Symposium, Proceedings July 2005, Page(s) 223- 226

Crake, D 2002, *An Embedded System for Short-Rage Wireless Digital Audio Communications*, Undergraduate thesis, Curtin University of WA

DallaVolta, C 2001, *Digital Audio Communications over an IrDA Link*, Undergraduate thesis, Curtin University of WA

Department of Training 2002, *Fund fosters Expertise through Innovation & Collaboration*, Internal report of Government of Western Australia, Page(s): 6

DLink 2007, *Dlink DPH-540 Wi-Fi Phone*, Retrieved: April 2007, from http://www.dlink.com/products/resource.asp?pid=485&rid=1991&sec=0

Dodds, I 1996, *A Wireless Stereo Link for Headsets*, Undergraduate thesis, Curtin University of WA

Fairhurst, G 2001, Digital Phase Lock Loop (DPLL), Retrieved: June 2007, from http://www.erg.abdn.ac.uk/users/gorry/course/phy-pages/dpll.html

Freedom Scientific 2009, *JAWS for Windows Screen Reading Software*, Retrieved: April 2009, from http://www.freedomscientific.com/products/fs/jaws-product-page.asp

Gizma 2006, *Wireless Stereo Headsets for Mobile Phones*, Retrieved: November 2006, from http://www.gizmag.com/go/4972/

Gratton, D 2007, Developing Practical Wireless Applications, Published by Digital Press, Page(s): 13-134

Hammer, F, Reichl, P, Nordstrom, T & Kubin, G 2003, *Corrupted Speech Data Considered Useful*, Retrieved: August 2004, from http://userver.ftw.at/~reichl/publications/AQS03.pdf

Hickey, M 2008, *Pogue shares tech support humor; We share Pogue*, Retrieved: June 2008, from http://www.crunchgear.com/2008/03/07/pogue-shares-tech-support-humor-we-share-pogue/

Hou, J & O'Brien, D 2006, Vertical *handover-decision-making algorithm using fuzzy logic for the integrated Radio-and-OW system*, IEEE Transactions on Wireless Communications, Volume 5, Issue 1, Jan. 2006, Page(s):176 - 185

Howitt, I 2003, *Mutual Interference Between Independent Bluetooth Piconets*, IEEE Transactions on Vehicular Technology, Volume 52, Issue 3, May 2003 Page(s): 708 - 718

Haasz, J 2006, *Withdrawal of the 802.15.3a PAR*, IEEE International Standards Programs and Governance, Retrieved: August 2006, from http://standards.ieee.org/board/nes/projects/802-15-3a.pdf

Holm, S, Hovind, O.B, Rostad, S & Holm, R 2005, *Indoors data communications using airborne ultrasound*, IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005. Proceedings, Volume 3, 18-23 March 2005 Page(s): 957 - 960

IBM 2000, *Things that talk: Using sound for device-to-device and device-to-human communication*, Retrieved: June 2003, from http://www.research.ibm.com/journal/sj/393/part1/gerasimov.html

Infrared Data Association 2001, "*Infrared Data Association Serial Infrared Physical Layer Specification*", Version 1.4, Infrared Data Association, California U.S.A

IrDA 2001, "*Infrared Data Association Serial Infrared Physical Layer Specification*", Version 1.4, Infrared Data Association, California U.S.A

IRLink 2004, *IR100 Infrared Wireless*, Retrieved: June 2004, from http://www.irlink.co.kr/ehtml/eheadset2.htm

Jivkova, S & Kavehrad, M.2003, *Shadowing and blockage in indoor optical wireless communications*, IEEE Global Telecommunications Conference, 2003, Volume: 6, Page(s): 3269- 3273

Linden, J 2003, *Finding Harmony Between VoIP and WLANs*, Retrieved: October 2005, from http://www.commsdesign.com/article/printableArticle.jhtml?articleID=16502471

Kowalk, W 2005, *The Dect Standard*, University of Oldenburg, Germany, Retrieved: June 2006, from http://einstein.informatik.uni-oldenburg.de/rechnernetze/seite24.htm

McCullen Melanie 1992, *Wireless Network Security*, Retrieved: June 2003, from , http://www.networkmagazine.com/article/NMG20000724S0062

Merritt, R & Mannion P 2004, *Wireless headset makers don't harmonize on stereo audio*, Retrieved: August 2004, from http://www.commsdesign.com/story/showArticle.jhtml?articleID=23901403

MobileBurn 2003, *Docker Magnetic Wireless Headset*, Retrieved: January 2004, from http://www.mobileburn.com/review.jsp?Page=1&Id=548

Moran Joseph 2002, *Wireless Home Networking, Part II - Wi-Fi Standards,* Retrieved: August 2003, from http://www.80211-planet.com/tutorials/article.php/1495031,

Murray, I & Pasquale, A 2006, *Headset Communications for Vision Impaired Persons in Multi-User Environments*, in ACM SIGACCESS Conference on Computers & Accessibility

Myslik, V 2003, Introduction to IrDA, Retrieved: June 2003, from http://hw-server.com/docs/irda.html

Ochi, H, Watanabe, Y, Shimura, T & Hattori, T 2008, *Experimental Results of Short Range Wideband Acoustic Communication Using QPSK and 8PSK*, OCEANS 2008 - MTS/IEEE Kobe Techno-Ocean, 8-11 April 2008, Page(s): 1 - 5

Ortiz, Luis 2001, *Solving QoS in VoIP: A Formula for Explosive Growth*, Brookout Technology, Needham MA USA, Page(s): 3

Netgear 2007, *WiFi Phone with Skype SPH200W*, Retrieved: April 2007, from http://www.netgear.com/Products/CommunicationsVoIP/Skype/SPH200W.aspx?detail=Specifications

Nist 2000, "*Commerce Department Announces Winner of Global Information Security Competition*", NIST, MD USA, Retrieved: July 2003 from, http://www.nist.gov/public_affairs/releases/g00-176.htm,

Nour-Eldin, A, Tolba, H & O'Shaughnessy, D 2004, *Automatic recognition of Bluetooth speech in 802.11 interference and the effectiveness of insertion-based compensation techniques*, IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). Volume 1, 17-21 May 2004 Page(s): 1033-1036

NPL 2006, *Acoustic Shock in Telephone and Headset Users, National Physics Laboratory*, UK, Retrieved: June 2006, from http://callcentreinfo.tmcnet.com/research/acousticshockdefinition.pdf

Ojala, P, Lakaniemi, A, Lepänaho, H & Jokimies, M 2006, *The Adaptive Multirate Speech Codec: System characteristics, quality, Advance and Deployment Strategies*, IEEE Communications Magazine, May 2006, Page(s): 59

Plantronics, 2006, *The CS361N Binaural SupraPlus Wireless Professional Headset System*. Retrieved: January 2007, from http://www.plantronics.com/australia/aus/products/office/wireless-headsets/supraplus-wireless-cs361n

Ramirez, R, Green, R 1999, *Indoor Optical Wireless Communications*, IEE Colloquium on 22 June 1999, Page(s): 14/1 - 14/7

Rodman, J 2006, *The effect of bandwidth on speech intelligibility*. Retrieved: January 2007, from http://www.polycom.com/common/documents/whitepapers/effect_of_bandwidth_on_speech_intelligibility_2.pdf

Sakurai, Y, Nishimaki, K, Toguchi, S & Sakane, M 2003, *A study of seamless communication method with the adequate switching between optical and RF wireless LAN*, in Dig. Tech. Papers. Int. Conf. Consumer Electronics Cat. No.03CH37416, Los Angeles, CA, 2003, Page(s): 264–265.

Spectrum IEEE 2003, *Making Unbreakable Code*, IEEE, New Jersey USA, Retrieved: July 2003, from Available: http://www.spectrum.ieee.org/WEBONLY/publicfeature/may02/code.html

The West Australian 2002, *Headset design opens jobs to blind people*, Article, 24th July 2002, Page(s): 36

Todd, A 2002, *Wireless Headset for Blind Computer Users*, Undergraduate thesis, Curtin University of WA

Truax, B. Handbook for Acoustic Ecology, Retrieved: October 2003, from http://www.sfu.ca/sonic-studio/

Tsai, S, Plumb & R, White, I 2004, *Indoor hybrid infrared-radio access network*, 12th International Symposium on Electron Devices for Microwave and Optoelectronic Applications, 8-9 Nov. 2004 Page(s):102 - 105

Waclawsky, J & Gunn, J 2004, *Solving the WLAN VOIP Challenge: Part 2*, Retrieved: August 2004, from http://www.commsdesign.com/design_corner/showArticle.jhtml;jsessionid=CBYLDXWWROKQAQSNDBGCKHQ?articleID=23900706

Wang, P, Jiang, H & Zhuang W 2006, *IEEE 802.11E Enhancement for Voice Service*, IEEE Communications magazine, Febuary 2006, Page(s): 30-35

Wave Report 2007, "*OFDM Tutorial*", Retrieved: March 2007, from http://www.wave-report.com/tutorials/OFDM.htm

Wikipedia 2008a, "*Frequency-hopping Spread spectrum*", Retrieved: October 2008, from http://en.wikipedia.org/wiki/Frequency-hopping_spread_spectrum

Wikipedia 2008b, "*Refreshable Braille display*”, Retrieved: April 2008, from http://en.wikipedia.org/wiki/Image:Refreshable_Braille_display.jpg)

Wikipedia 2008c, "*Modulation*", Retrieved: October 2008, from http://en.wikipedia.org/wiki/Modulation

Wikipedia 2008d, "*Channel Access Method*", Retrieved: October 2008, from http://en.wikipedia.org/wiki/Channel_access_method

Xue, J & Todd, T 2001, *Basestation collaboration in Bluetooth voice networks*, IEEE International Conference on Local Computer Networks, 2001, Proceedings. 14-16 Nov 2001 Page(s): 533 - 538

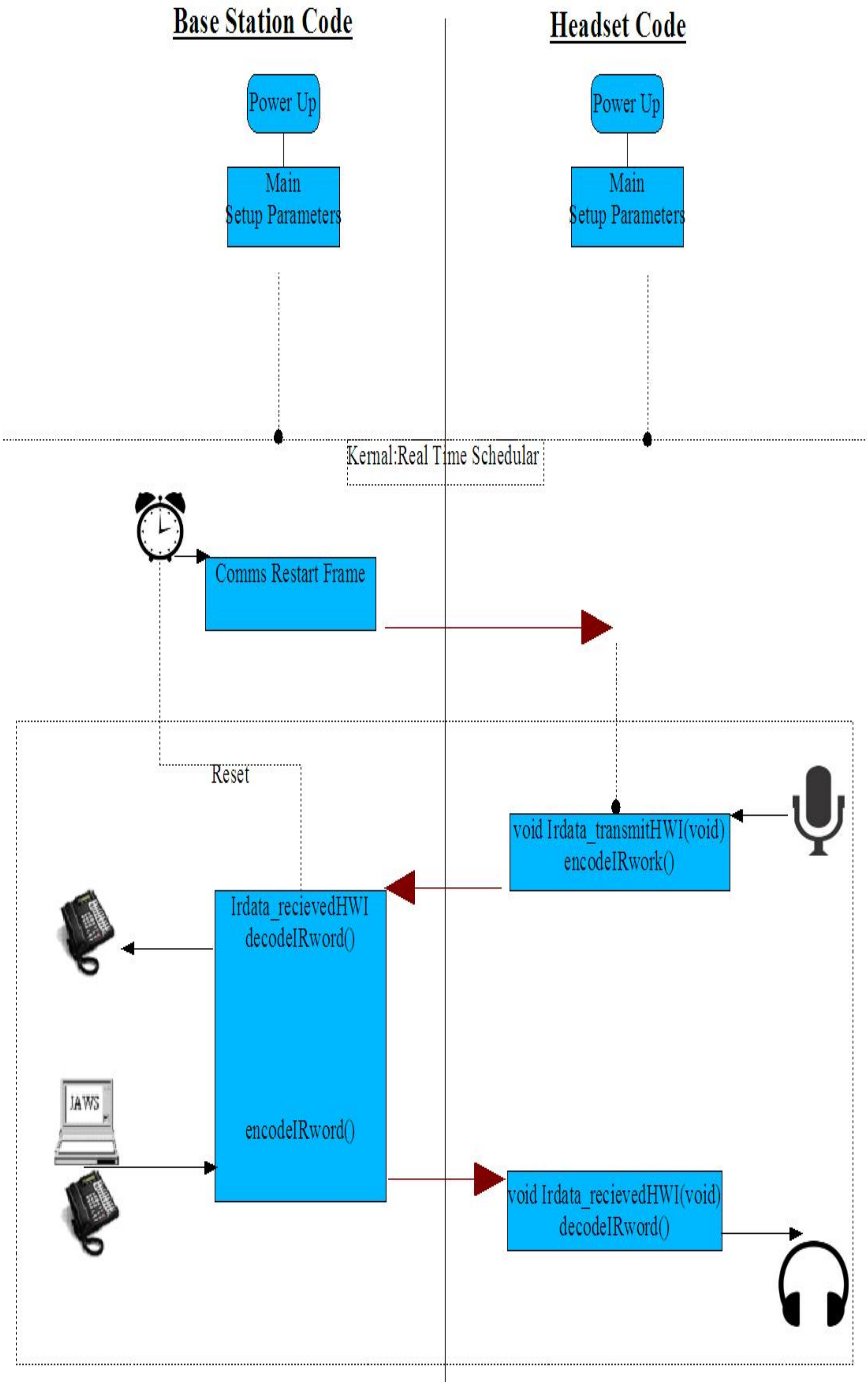
Xydis, T & Wilson, S 2002, *Security Comparison: Bluetooth^{TM} Communications vs. 802.11*, Bluetooth Security Experts Group

Zhengzhong, W, Xianxin, Z, Wenge, Y & Zilin, L 2004, *System level simulation modeling of Bluetooth voice and its interference*, Proceedings ICSP '04, 7th International Conference on Signal Processing, Volume 1, Page(s):29 - 32

Zurbes, S, Stahl, W, Matheus, K & Haartsen J 2000, *Radio Network Performance of Bluetooth*, IEEE International Conference on Communications, Volume 3, 18-22 June 2000 Page(s):1563 – 1567

Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.

Appendix I. Software Flow Chart



Appendix II.   Source Code

Application Software - Texas Instruments Code Composer Studio IDE v2.1.

II.a.   Headset Source Code

```
/*5502 IR Headset Code
2005-08-25: Cleaned up code
Andrew Pasquale, Curtin University of Technology*/

#define CHIP_5502 1

#include <log.h>
#include <clk.h>
#include <hwi.h>
#include <csl_mcbsp.h>
#include <csl_irq.h>
#include <csl_pll.h>
#include <csl_emif.h>
#include <csl_chip.h>
#include <intrindefs.h>
#include <csl_dma.h>
#include "clktestcfg.h"

void IRdata_recievedHWI(void);
void IRdata_transmitHWI(void);

MCBSP_Handle mhMcbsp, C55XX_CONTROLHANDLE_hMcbsp, C55XX_DMA_MCBSP_hMcbsp;

#define N       4

/* Define transmit and receive buffers */
#pragma DATA_SECTION(xmt1,"dmaMem")
Int16 xmt1[6];
#pragma DATA_SECTION(xmt2,"dmaMem")
Int16 xmt2[6];

#pragma DATA_SECTION(MicSample,"dmaMem")
Int16 MicSample[7];

#define my_AIC23_RESET       15
#define my_AIC23_NUMREGS     10
typedef int my_AIC23_CodecHandle;
my_AIC23_CodecHandle hCodec;

typedef struct my_AIC23_Config {               /* Parameter Structure for the AIC23 Codec */
   int regs[my_AIC23_NUMREGS];
} my_AIC23_Config;

#define my_AIC23_DEFAULTCONFIG { \
   0x0017, /* Set-Up Reg 0       Left line input channel volume control */  \
         /* LRS     0          simultaneous left/right volume: disabled */\
         /* LIM     0          left line input mute: disabled */          \
         /* XX      00         reserved */                                \
         /* LIV     10111      left line input volume: 0 dB */            \
                                                                          \
   0x0017, /* Set-Up Reg 1       Right line input channel volume control */ \
         /* RLS     0          simultaneous right/left volume: disabled */\
         /* RIM     0          right line input mute: disabled */         \
         /* XX      00         reserved */                                \
         /* RIV     10111      right line input volume: 0 dB */           \
                                                                          \
   0x01f9, /* Set-Up Reg 2       Left channel headphone volume control */   \
         /* LRS     1          simultaneous left/right volume: enabled */ \
         /* LZC     1          left channel zero-cross detect: enabled */ \
         /* LHV     1111001    left headphone volume: 0 dB */             \
                                                                          \
   0x01f9, /* Set-Up Reg 3       Right channel headphone volume control */  \
         /* RLS     1          simultaneous right/left volume: enabled */ \
         /* RZC     1          right channel zero-cross detect: enabled */\
         /* RHV     1111001    right headphone volume: 0 dB */           \
                                                                          \
   0x0011, /* Set-Up Reg 4       Analog audio path control */         \
         /* X       0          reserved */                               \
         /* STA     00         sidetone attenuation: -6 dB */            \
         /* STE     0          sidetone: disabled */                     \
         /* DAC     1          DAC: selected */                          \
         /* BYP     0          bypass: off */                            \
         /* INSEL   0          input select for ADC: line */             \
         /* MICM    0          microphone mute: disabled */              \
         /* MICB    1          microphone boost: enabled */              \
                                                                          \
   0x0000, /* Set-Up Reg 5       Digital audio path control */        \
         /* XXXXX   00000      reserved */                               \
         /* DACM    0          DAC soft mute: disabled */                \
         /* DEEMP   00         deemphasis control: disabled */           \
         /* ADCHP   0          ADC high-pass filter: disabled */         \
                                                                          \
   0x0000, /* Set-Up Reg 6       Power down control */                \
         /* X       0          reserved */                               \
         /* OFF     0          device power: on (i.e. not off) */        \
         /* CLK     0          clock: on */                              \
         /* OSC     0          oscillator: on */                         \
         /* OUT     0          outputs: on */                            \
         /* DAC     0          DAC: on */                                \
         /* ADC     0          ADC: on */                                \
         /* MIC     0          microphone: on */                         \
         /* LINE    0          line input: on */                         \
                                                                          \
   0x0043, /* Set-Up Reg 7       Digital audio interface format */    \
         /* XX      00         reserved */                               \
         /* MS      1          master/slave mode: master */              \
         /* LRSWAP  0          DAC left/right swap: disabled */          \
         /* LRP     0          DAC lrp: MSB on 1st BCLK */               \
         /* IWL     00         input bit length: 16 bit */               \
         /* FOR     11         data format: DSP format */                \
                                                                          \
   0x0081, /* Set-Up Reg 8       Sample rate control */               \
         /* X       0          reserved */                               \
         /* CLKOUT  1          clock output divider: 2 (MCLK/2) */       \
         /* CLKIN   0          clock input divider: 2 (MCLK/2) */        \
         /* SR,BOSR 00000      sampling rate: ADC  48 kHz DAC  48 kHz */ \
         /* USB/N   1          clock mode select (USB/normal): USB */    \
                                                                          \
   0x0001  /* Set-Up Reg 9       Digital interface activation */      \
         /* XX..X   00000000   reserved */                               \
         /* ACT     1          active */                                 \
}


/* Codec configuration settings */
my_AIC23_Config config = {
   0x0097,  // 0 DSK5510_AIC23_LEFTINVOL  Left line input channel volume
   0x0097,  // 1 DSK5510_AIC23_RIGHTINVOL Right line input channel volume
   0x01E9,  // 2 DSK5510_AIC23_LEFTHPVOL  Left channel headphone volume
   0x01E9,  // 3 DSK5510_AIC23_RIGHTHPVOL Right channel headphone volume
   0x00D5,  // 4 DSK5510_AIC23_ANAPATH    Analog audio path control, line input
   0x0001,  // 5 DSK5510_AIC23_DIGPATH    Digital audio path control, with high pass filter
   0x0061,  // 6 DSK5510_AIC23_POWERDOWN  Power down control, turn off oscillator, line in, clkout
   0x0043,  // 7 DSK5510_AIC23_DIGIF      Digital audio interface format
   0x0098,  // 8 DSK5510_AIC23_SAMPLERATE Sample rate control, 32KHz but since codec is at 3 MHz, is really 8KHz
   0x0001   // 9 DSK5510_AIC23_DIGACT     Digital interface activation
};

PLL_Config MyPLLConfig = {
                0x1, /* PLLCSR */
                0x2, /* PLLM */
                0x8000, /* PLLDIV0 */
                0x8000, /* PLLDIV1 */
                0x8000, /* PLLDIV2 */
                0x8001, /* PLLDIV3 */
                0x0, /* OSCDIV1 */
                0x0, /* WAKEUP */
                0x1, /* CLKMD */
                0x2 /* CLKOUTSR */
```

```
};

MCBSP_Config ConfigMcbsp1 = {
0x1000, /* spcr1 0001 0000 0000 0000, SPI (clock stop) mode enabled*/
0x0100, /* spcr2 */
0x0000, /* rcr1 */
0x0000, /* rcr2 */
0x0040, /* xcr1 */
0x0002, /* xcr2 */
0x000C, /* srgr1 0000 0000 0110 0011*/
0x2013, /* srgr2 */
0x0000, /* mcr1 */
0x0000, /* mcr2 */
0x0A0A, /* pcr, 0001 1010 0000 1010 */
0xFFFF, /* rcera */
0xFFFF, /* rcerb */
0xFFFF, /* rcerc */
0xFFFF, /* rcerd */
0xFFFF, /* rcere */
0xFFFF, /* rcerf */
0xFFFF, /* rcerg */
0xFFFF, /* rcerh */
0xFFFF, /* xcera */
0xFFFF, /* xcerb */
0xFFFF, /* xcerc */
0xFFFF, /* xcerd */
0xFFFF, /* xcere */
0xFFFF, /* xcerf */
0xFFFF, /* xcerg */
0xFFFF /* xcerh */
};

MCBSP_Config ConfigMcbsp2 = {
0x0000, /* spcr1 */
0x0100, /* spcr2 */
0x0140, /* rcr1 */
0x0000, /* rcr2 ,make rdatdly = 1, check new headset board, if 0 works since overflow can occur if user speaks loud*/
0x01A0, /* xcr1, 32bit word for transmit */
0x0000, /* xcr2 */
0x0000, /* srgr1 */
0x2000, /* srgr2 */
0x0000, /* mcr1 */
0x0000, /* mcr2 */
0x0182, /* pcr, make clock polarity negative edge triggered, solves mic overflow problem */
0xFFFF, /* rcera */
0xFFFF, /* rcerb */
0xFFFF, /* rcerc */
0xFFFF, /* rcerd */
0xFFFF, /* rcere */
0xFFFF, /* rcerf */
0xFFFF, /* rcerg */
0xFFFF, /* rcerh */
0xFFFF, /* xcera */
0xFFFF, /* xcerb */
0xFFFF, /* xcerc */
0xFFFF, /* xcerd */
0xFFFF, /* xcere */
0xFFFF, /* xcerf */
0xFFFF, /* xcerg */
0xFFFF /* xcerh */
};

/* Create a MCBSP configuration structure */
  MCBSP_Config ConfigLoopBack320= {
  MCBSP_SPCR1_RMK(
    MCBSP_SPCR1_DLB_OFF,                  /* DLB    = 0 */
    MCBSP_SPCR1_RJUST_RZF,                /* RJUST  = 0 */
    MCBSP_SPCR1_CLKSTP_DISABLE,           /* CLKSTP = 0 */
    MCBSP_SPCR1_DXENA_NA,                 /* DXENA  = 0 */
    MCBSP_SPCR1_ABIS_DISABLE,             /* ABIS   = 0 */
    MCBSP_SPCR1_RINTM_RRDY,               /* RINTM  = 0 */
    0,                            /* RSYNCER = 0 */
    MCBSP_SPCR1_RRST_DISABLE              /* RRST   = 0 */
   ),
    MCBSP_SPCR2_RMK(
    MCBSP_SPCR2_FREE_NO,                  /* FREE   = 0 */
    MCBSP_SPCR2_SOFT_NO,                    /* SOFT   = 0 */
    MCBSP_SPCR2_FRST_FSG,                   /* FRST   = 1 */
    MCBSP_SPCR2_GRST_CLKG,                    /* GRST   = 1 */
    MCBSP_SPCR2_XINTM_XRDY,                  /* XINTM  = 0 */
    0,                              /* XSYNCER = N/A */
    MCBSP_SPCR2_XRST_DISABLE                 /* XRST   = 0 */
   ),
    MCBSP_RCR1_RMK(
  MCBSP_RCR1_RFRLEN1_OF(16),                 /* RFRLEN1 = 3 */
  MCBSP_RCR1_RWDLEN1_32BIT                   /* RWDLEN1 = 5 */
  ),
  MCBSP_RCR2_RMK(
    MCBSP_RCR2_RPHASE_SINGLE,                /* RPHASE  = 0 */
    MCBSP_RCR2_RFRLEN2_OF(0),              /* RFRLEN2 = 0 */
    MCBSP_RCR2_RWDLEN2_8BIT,                /* RWDLEN2 = 0 */
    MCBSP_RCR2_RCOMPAND_MSB,                 /* RCOMPAND = 0 */
    MCBSP_RCR2_RFIG_NO,                    /* RFIG    = 1 */
    MCBSP_RCR2_RDATDLY_1BIT                 /* RDATDLY = 1 */
    ),
    MCBSP_XCR1_RMK(
    MCBSP_XCR1_XFRLEN1_OF(5),              /* XFRLEN1 = 1 */
    MCBSP_XCR1_XWDLEN1_32BIT                /* XWDLEN1 = 5 */

  ),
  MCBSP_XCR2_RMK(
    MCBSP_XCR2_XPHASE_SINGLE,                /* XPHASE  = 0 */
    MCBSP_XCR2_XFRLEN2_OF(0),              /* XFRLEN2 = 0 */
    MCBSP_XCR2_XWDLEN2_8BIT,                /* XWDLEN2 = 0 */
    MCBSP_XCR2_XCOMPAND_MSB,                 /* XCOMPAND = 0 */
    MCBSP_XCR2_XFIG_NO,                    /* XFIG    = 1 */
    MCBSP_XCR2_XDATDLY_0BIT                 /* XDATDLY = 0 */
  ),
  MCBSP_SRGR1_RMK(
    MCBSP_SRGR1_FWID_OF(1),                /* FWID    = 1 */
    MCBSP_SRGR1_CLKGDV_OF(2)                /* CLKGDV  = 2 */
  ),
  MCBSP_SRGR2_RMK(
    MCBSP_SRGR2_GSYNC_SYNC,                 /* FREE    = 1 */
    MCBSP_SRGR2_CLKSP_RISING,              /* CLKSP   = 0 */
    MCBSP_SRGR2_CLKSM_INTERNAL,             /* CLKSM   = 1 */
    MCBSP_SRGR2_FSGM_DXR2XSR,              /* FSGM    = 0 */
    MCBSP_SRGR2_FPER_OF(15)               /* FPER    = 15 */
  ),
  MCBSP_MCR1_DEFAULT,
  MCBSP_MCR2_DEFAULT,
  MCBSP_PCR_RMK(
    MCBSP_PCR_XIOEN_SP,                    /* XIOEN    = 0   */
    MCBSP_PCR_RIOEN_SP,                    /* RIOEN    = 0   */
    MCBSP_PCR_FSXM_INTERNAL,                /* FSXM     = 1   */
    MCBSP_PCR_FSRM_EXTERNAL,                /* FSRM     = 0   */
    MCBSP_PCR_CLKXM_OUTPUT,                 /* CLKXM    = 1   */
    MCBSP_PCR_CLKRM_OUTPUT,                 /* CLKRM    = 1   */
    0,                              /* DXSTAT = N/A   */
    MCBSP_PCR_SCLKME_NO,                   /* SCLKME   = 0   */
    MCBSP_PCR_FSXP_ACTIVEHIGH,              /* FSXP     = 0   */
    MCBSP_PCR_FSRP_ACTIVEHIGH,              /* FSRP     = 0   */
    MCBSP_PCR_CLKXP_RISING,                /* CLKXP    = 0   */
    MCBSP_PCR_CLKRP_RISING                /* CLKRP    = 1   */
  ),
  MCBSP_RCERA_DEFAULT,
  MCBSP_RCERB_DEFAULT,
  MCBSP_RCERC_DEFAULT,
  MCBSP_RCERD_DEFAULT,
  MCBSP_RCERE_DEFAULT,
  MCBSP_RCERF_DEFAULT,
  MCBSP_RCERG_DEFAULT,
  MCBSP_RCERH_DEFAULT,
  MCBSP_XCERA_DEFAULT,
  MCBSP_XCERB_DEFAULT,
  MCBSP_XCERC_DEFAULT,
  MCBSP_XCERD_DEFAULT,
  MCBSP_XCERE_DEFAULT,
  MCBSP_XCERF_DEFAULT,
  MCBSP_XCERG_DEFAULT,
  MCBSP_XCERH_DEFAULT
};
```

```
/* Create DMA Transmit Side Configuration */
DMA_Config  dmaRcvConfig = {
  DMA_DMACSDP_RMK(
    DMA_DMACSDP_DSTBEN_NOBURST,
    DMA_DMACSDP_DSTPACK_OFF,
    DMA_DMACSDP_DST_DARAMPORT1,
    DMA_DMACSDP_SRCBEN_NOBURST,
    DMA_DMACSDP_SRCPACK_OFF,
    DMA_DMACSDP_SRC_PERIPH,
    DMA_DMACSDP_DATATYPE_16BIT
  ),                                /* DMACSDP  */
  DMA_DMACCR_RMK(
    DMA_DMACCR_DSTAMODE_POSTINC,
    DMA_DMACCR_SRCAMODE_CONST,
    DMA_DMACCR_ENDPROG_OFF,
    DMA_DMACCR_WP_ENABLE,
    DMA_DMACCR_REPEAT_ALWAYS,
    DMA_DMACCR_AUTOINIT_ON,
    DMA_DMACCR_EN_STOP,
    DMA_DMACCR_PRIO_HI,
    DMA_DMACCR_FS_DISABLE,
    DMA_DMACCR_SYNC_REVT2
  ),                                /* DMACCR   */
  DMA_DMACICR_RMK(
    DMA_DMACICR_BLOCKIE_ON,
    DMA_DMACICR_LASTIE_OFF,
    DMA_DMACICR_FRAMEIE_OFF,
    DMA_DMACICR_FIRSTHALFIE_OFF,
    DMA_DMACICR_DROPIE_OFF,
    DMA_DMACICR_TIMEOUTIE_OFF
  ),                                /* DMACICR  */
    (DMA_AdrPtr)(MCBSP_ADDR(DRR12)),                    /* DMACSSAL */
    0,                              /* DMACSSAU */
    (DMA_AdrPtr)&MicSample,                             /* DMACDSAL */
    0,                              /* DMACDSAU */
    2,                              /* DMACEN   */
    3,                              /* DMACFN   */
    0,                                      /* DMACSFI  */
    0,                                              /* DMACSEI  */
    0,                                      /* DMACDFI  */
    0                                       /* DMACDEI  */
};

DMA_Config  dmaXmtConfig2 = {
  DMA_DMACSDP_RMK(
    DMA_DMACSDP_DSTBEN_NOBURST,
    DMA_DMACSDP_DSTPACK_OFF,
    DMA_DMACSDP_DST_PERIPH,
    DMA_DMACSDP_SRCBEN_NOBURST,
    DMA_DMACSDP_SRCPACK_OFF,
    DMA_DMACSDP_SRC_DARAMPORT0,
    DMA_DMACSDP_DATATYPE_16BIT
  ),                                /* DMACSDP  */
  DMA_DMACCR_RMK(
    DMA_DMACCR_DSTAMODE_CONST,
    DMA_DMACCR_SRCAMODE_POSTINC,
    DMA_DMACCR_ENDPROG_OFF,
    DMA_DMACCR_WP_DEFAULT,
    DMA_DMACCR_REPEAT_ALWAYS,
    DMA_DMACCR_AUTOINIT_ON,
    DMA_DMACCR_EN_STOP,
    DMA_DMACCR_PRIO_HI,
    DMA_DMACCR_FS_DISABLE,
    DMA_DMACCR_SYNC_XEVT2
  ),                                /* DMACCR   */
  DMA_DMACICR_RMK(
    DMA_DMACICR_BLOCKIE_OFF,
    DMA_DMACICR_LASTIE_OFF,
    DMA_DMACICR_FRAMEIE_OFF,
    DMA_DMACICR_FIRSTHALFIE_OFF,
    DMA_DMACICR_DROPIE_OFF,
    DMA_DMACICR_TIMEOUTIE_OFF
  ),                                /* DMACICR  */
    (DMA_AdrPtr)&xmt2[0],                   /* DMACSSAL */
    0,                              /* DMACSSAU */
    (DMA_AdrPtr)(MCBSP_ADDR(DXR22)),        /* DMACDSAL */
    0,                              /* DMACDSAU */
    2,                              /* DMACEN   */
    3,                              /* DMACFN   */
    1,                                      /* DMACSFI  */
    1,                                              /* DMACSEI  */
    0,                                      /* DMACDFI  */
    0                                       /* DMACDEI  */
};


DMA_Config  dmaXmtConfig1 = {
  DMA_DMACSDP_RMK(
    DMA_DMACSDP_DSTBEN_NOBURST,
    DMA_DMACSDP_DSTPACK_OFF,
    DMA_DMACSDP_DST_PERIPH,
    DMA_DMACSDP_SRCBEN_NOBURST,
    DMA_DMACSDP_SRCPACK_OFF,
    DMA_DMACSDP_SRC_DARAMPORT1,
    DMA_DMACSDP_DATATYPE_16BIT
  ),                              /* DMACSDP  */
  DMA_DMACCR_RMK(
    DMA_DMACCR_DSTAMODE_CONST,
    DMA_DMACCR_SRCAMODE_POSTINC,
    DMA_DMACCR_ENDPROG_OFF,
    DMA_DMACCR_WP_DEFAULT,
    DMA_DMACCR_REPEAT_ALWAYS,
    DMA_DMACCR_AUTOINIT_ON,
    DMA_DMACCR_EN_STOP,
    DMA_DMACCR_PRIO_LOW,
    DMA_DMACCR_FS_DISABLE,
    DMA_DMACCR_SYNC_XEVT2
  ),                              /* DMACCR   */
  DMA_DMACICR_RMK(
    DMA_DMACICR_BLOCKIE_OFF,
    DMA_DMACICR_LASTIE_OFF,
    DMA_DMACICR_FRAMEIE_OFF,
    DMA_DMACICR_FIRSTHALFIE_OFF,
    DMA_DMACICR_DROPIE_OFF,
    DMA_DMACICR_TIMEOUTIE_OFF
  ),                              /* DMACICR  */
    (DMA_AdrPtr)&xmt1[0],                   /* DMACSSAL */
    0,                              /* DMACSSAU */
    (DMA_AdrPtr)(MCBSP_ADDR(DXR12)),        /* DMACDSAL */
    0,                              /* DMACDSAU */
    2,                              /* DMACEN   */
    3,                              /* DMACFN   */
    1,                                      /* DMACSFI  */
    1,                                              /* DMACSEI  */
    0,                                      /* DMACDFI  */
    0                                       /* DMACDEI  */
};


/* Define a DMA_Handle object to be used with DMA_open function */
DMA_Handle hDmaRcv, hDmaXmt1, hDmaXmt2;
Uint16 srcAddrHi, srcAddrLo;
Uint16 dstAddrHi, dstAddrLo;
Uint16 xmtEventId, rcvEventId;
Uint16 old_intm, dmastat1, dmastat2;

Int16 Headset_code=0x34;
/* Internal codec state used to simulate read/write functionality */
static my_AIC23_Config codecstate = my_AIC23_DEFAULTCONFIG;
Int16       HdPhoneSampleL[N], HdPhoneSampleR[N], LPHdPhoneSampleL[N], MicSamplei[8], tempTx, tempRcv,
                        LPHdPhoneSampleR[N],HdPhoneSampleLold[3], HdPhoneSampleRold[3], xmtoldL, xmtoldR, CdChk,
                        delay, TxIrCode;
Uint32 recieved_wordLlsb[N], recieved_wordRlsb[N], recieved_wordLmsb[N], recieved_wordRmsb[N],
                transmit_wordMsb[N], transmit_wordLsb[N], dummy, ErrCode, RcvCode, BaseIrCode;
Uint32 regval, regnum;
Int32 HdPhone;
Uint16 RecvEventId0, RecvEventId2, DmaRcvEventId, ECshrt;
int i, j, playevent = 0, token = 0, sync = 0, syncL = 0, syncR = 0,x, tag,
            different_base, correct_base_LOS;
Uint32 addr;
```

```
Uchar crc8_codeRcv, indexRcv, crc8_codeTx, indexTx;
Uint16 limit = 31000, sample;                          //limit of audio volume variation

Uchar crc8_data[] = { //CRC checksum lookup table
0x00, 0x5e, 0xbc, 0xe2, 0x61, 0x3f, 0xdd, 0x83,
0xc2, 0x9c, 0x7e, 0x20, 0xa3, 0xfd, 0x1f, 0x41,
0x9d, 0xc3, 0x21, 0x7f, 0xfc, 0xa2, 0x40, 0x1e,
0x5f, 0x01, 0xe3, 0xbd, 0x3e, 0x60, 0x82, 0xdc,
0x23, 0x7d, 0x9f, 0xc1, 0x42, 0x1c, 0xfe, 0xa0,
0xe1, 0xbf, 0x5d, 0x03, 0x80, 0xde, 0x3c, 0x62,
0xbe, 0xe0, 0x02, 0x5c, 0xdf, 0x81, 0x63, 0x3d,
0x7c, 0x22, 0xc0, 0x9e, 0x1d, 0x43, 0xa1, 0xff,
0x46, 0x18, 0xfa, 0xa4, 0x27, 0x79, 0x9b, 0xc5,
0x84, 0xda, 0x38, 0x66, 0xe5, 0xbb, 0x59, 0x07,
0xdb, 0x85, 0x67, 0x39, 0xba, 0xe4, 0x06, 0x58,
0x19, 0x47, 0xa5, 0xfb, 0x78, 0x26, 0xc4, 0x9a,
0x65, 0x3b, 0xd9, 0x87, 0x04, 0x5a, 0xb8, 0xe6,
0xa7, 0xf9, 0x1b, 0x45, 0xc6, 0x98, 0x7a, 0x24,
0xf8, 0xa6, 0x44, 0x1a, 0x99, 0xc7, 0x25, 0x7b,
0x3a, 0x64, 0x86, 0xd8, 0x5b, 0x05, 0xe7, 0xb9,
0x8c, 0xd2, 0x30, 0x6e, 0xed, 0xb3, 0x51, 0x0f,
0x4e, 0x10, 0xf2, 0xac, 0x2f, 0x71, 0x93, 0xcd,
0x11, 0x4f, 0xad, 0xf3, 0x70, 0x2e, 0xcc, 0x92,
0xd3, 0x8d, 0x6f, 0x31, 0xb2, 0xec, 0x0e, 0x50,
0xaf, 0xf1, 0x13, 0x4d, 0xce, 0x90, 0x72, 0x2c,
0x6d, 0x33, 0xd1, 0x8f, 0x0c, 0x52, 0xb0, 0xee,
0x32, 0x6c, 0x8e, 0xd0, 0x53, 0x0d, 0xef, 0xb1,
0xf0, 0xae, 0x4c, 0x12, 0x91, 0xcf, 0x2d, 0x73,
0xca, 0x94, 0x76, 0x28, 0xab, 0xf5, 0x17, 0x49,
0x08, 0x56, 0xb4, 0xea, 0x69, 0x37, 0xd5, 0x8b,
0x57, 0x09, 0xeb, 0xb5, 0x36, 0x68, 0x8a, 0xd4,
0x95, 0xcb, 0x29, 0x77, 0xf4, 0xaa, 0x48, 0x16,
0xe9, 0xb7, 0x55, 0x0b, 0x88, 0xd6, 0x34, 0x6a,
0x2b, 0x75, 0x97, 0xc9, 0x4a, 0x14, 0xf6, 0xa8,
0x74, 0x2a, 0xc8, 0x96, 0x15, 0x4b, 0xa9, 0xf7,
0xb6, 0xe8, 0x0a, 0x54, 0xd7, 0x89, 0x6b, 0x35,
};

volatile ioport unsigned int* SPCR1;
volatile ioport unsigned int* SPCR2;
volatile ioport unsigned int* SRGR1;
volatile ioport unsigned int* SRGR2;
volatile ioport unsigned int* PCR;

/***************************************
 * Set the Codec configuration registers
 ***************************************/
void my_AIC23_rset(int hCodec, Uint16 regnum, Uint16 regval)
{
   /* Mask off lower 9 bits */
   regval &= 0x1ff;

   /* Wait for XRDY signal before writing data to DXR */
   while (!MCBSP_xrdy(C55XX_CONTROLHANDLE_hMcbsp));

   /* Write 16 bit data value to DXR */
   MCBSP_write16(C55XX_CONTROLHANDLE_hMcbsp, (regnum << 9) | regval);

   /* Save register value if regnum is in range */
   if (regnum < my_AIC23_NUMREGS)
      codecstate.regs[regnum] = regval;
}

/***************************************
 *  Configurate the Codec
 ***************************************/
void my_AIC23_config(int hCodec, my_AIC23_Config *Config)
{
   int i;

   /* Use default parameters if none are given */
   if (Config == NULL)
          Config = &config;

   /* Assign each register */
   for (i = 0; i < my_AIC23_NUMREGS; i++)
```

```
        my_AIC23_rset(hCodec, i, Config -> regs[i]);
}
/****************************************
 *  Write a 32-bit value to the codec
 ****************************************/
CSLBool my_AIC23_write32(Int32 val)
{
    /* Wait for XRDY signal before writing data to DXR */
    if(!MCBSP_xrdy(C55XX_DMA_MCBSP_hMcbsp)) {
        return (FALSE);
    }

    /* Write 32 bit data value to DXR, shift to match format mode */
    MCBSP_write32(C55XX_DMA_MCBSP_hMcbsp, val);

    return (TRUE);
}
/****************************************
 *  Write a 16-bit value to the codec
 ****************************************/
CSLBool my_AIC23_write16(Int16 val)
{
    /* Wait for XRDY signal before writing data to DXR */
    if(!MCBSP_xrdy(C55XX_DMA_MCBSP_hMcbsp)) {
        return (FALSE);
    }

    /* Write 32 bit data value to DXR, shift to match format mode */
    MCBSP_write16(C55XX_DMA_MCBSP_hMcbsp, val);

    return (TRUE);
}
/****************************************
 *  Write a 16-bit valueto the codec
 ****************************************/
CSLBool my_AIC23_read16(Int16 *val)
{
    /* Wait until data is ready then read */
    if (!MCBSP_rrdy(C55XX_DMA_MCBSP_hMcbsp)) {
        return (FALSE);
    }

    /* Read the data */
    *val = MCBSP_read16(C55XX_DMA_MCBSP_hMcbsp);
    return (TRUE);
}
/********************************************************************************
  ----------------------decode IR word into audio sample  --------------------
 ********************************************************************************/
void decodeIRword(Int16 *AudioSample, Uint32 recieved_wordLSB, Uint32 recieved_wordMSB)
{
Int16         HdPhoneSample = 0x0000;

              if ((recieved_wordLSB & 0x00000007) != 0) {HdPhoneSample |= 0x0001;}
              if ((recieved_wordLSB & 0x00000070) != 0) {HdPhoneSample |= 0x0002;}
              if ((recieved_wordLSB & 0x00000700) != 0) {HdPhoneSample |= 0x0004;}
              if ((recieved_wordLSB & 0x00007000) != 0) {HdPhoneSample |= 0x0008;}
              if ((recieved_wordLSB & 0x00070000) != 0) {HdPhoneSample |= 0x0010;}
              if ((recieved_wordLSB & 0x00700000) != 0) {HdPhoneSample |= 0x0020;}
              if ((recieved_wordLSB & 0x07000000) != 0) {HdPhoneSample |= 0x0040;}
              if ((recieved_wordLSB & 0x70000000) != 0) {HdPhoneSample |= 0x0080;}

              if ((recieved_wordMSB & 0x00000007) != 0) {HdPhoneSample |= 0x0100;}
              if ((recieved_wordMSB & 0x00000070) != 0) {HdPhoneSample |= 0x0200;}
              if ((recieved_wordMSB & 0x00000700) != 0) {HdPhoneSample |= 0x0400;}
              if ((recieved_wordMSB & 0x00007000) != 0) {HdPhoneSample |= 0x0800;}
              if ((recieved_wordMSB & 0x00070000) != 0) {HdPhoneSample |= 0x1000;}
              if ((recieved_wordMSB & 0x00700000) != 0) {HdPhoneSample |= 0x2000;}
              if ((recieved_wordMSB & 0x07000000) != 0) {HdPhoneSample |= 0x4000;}
              if ((recieved_wordMSB & 0x70000000) != 0) {HdPhoneSample |= 0x8000;}
              *AudioSample = HdPhoneSample;
}

/********************************************************************************
  ----------------------encode audio sample into IR word  --------------------
 ********************************************************************************/
```

```
void encodeIRword(Int16 MicSample, Uint32 *transmit_wordLSB, Uint32 *transmit_wordMSB)
{
Uint32        IRwordMSB = 0x00000000, IRwordLSB = 0x00000000;

              if ((MicSample & 0x0001) != 0) {IRwordLSB |= 0x00000002;}
              if ((MicSample & 0x0002) != 0) {IRwordLSB |= 0x00000020;}
              if ((MicSample & 0x0004) != 0) {IRwordLSB |= 0x00000200;}
              if ((MicSample & 0x0008) != 0) {IRwordLSB |= 0x00002000;}
              if ((MicSample & 0x0010) != 0) {IRwordLSB |= 0x00020000;}
              if ((MicSample & 0x0020) != 0) {IRwordLSB |= 0x00200000;}
              if ((MicSample & 0x0040) != 0) {IRwordLSB |= 0x02000000;}
              if ((MicSample & 0x0080) != 0) {IRwordLSB |= 0x20000000;}

              if ((MicSample & 0x0100) != 0) {IRwordMSB |= 0x00000002;}
              if ((MicSample & 0x0200) != 0) {IRwordMSB |= 0x00000020;}
              if ((MicSample & 0x0400) != 0) {IRwordMSB |= 0x00000200;}
              if ((MicSample & 0x0800) != 0) {IRwordMSB |= 0x00002000;}
              if ((MicSample & 0x1000) != 0) {IRwordMSB |= 0x00020000;}
              if ((MicSample & 0x2000) != 0) {IRwordMSB |= 0x00200000;}
              if ((MicSample & 0x4000) != 0) {IRwordMSB |= 0x02000000;}
              if ((MicSample & 0x8000) != 0) {IRwordMSB |= 0x20000000;}

              *transmit_wordLSB = IRwordLSB;
   *transmit_wordMSB = IRwordMSB;
}


/**************************************************************************************
 *------------------------Send IR transmission,from DMA2 block complete interrupts---
 *************************************************************************************/
void IRdata_transmitHWI(void)
{
              //--------------------Send Headset Audio Sample to Base Station--------------
              asm(" NOP");
              asm(" NOP");      // Put MCBSP reciever into reset, and prevent noise interference
              *SPCR1 = 0x0000;  // from disrupting the protocol
              asm(" NOP");
              asm(" NOP");
              *PCR = 0x0A00;
              asm(" NOP");
              asm(" NOP");
//            asm("         BSET XF  ; set xf pin");

              MicSamplei[0] = MicSample[1];//Re-enable DMA routine ASAP to prevent stalling
   MicSamplei[1] = MicSample[3];
   MicSamplei[2] = MicSample[5];
              DMA_RGETH(hDmaRcv, DMACSR);

              //correct_base_LOS = 0;              //force condition for testing
              if (correct_base_LOS < 4) {

                            transmit_wordMsb[0] &= 0x22222222;
                            transmit_wordLsb[0] &= 0x22222222;
                            transmit_wordMsb[1] &= 0x22222222;
                            transmit_wordLsb[1] &= 0x22222222;
                            transmit_wordMsb[2] &= 0x22222222;
                            transmit_wordLsb[2] &= 0x22222222;

                            while (!MCBSP_xrdy(mhMcbsp));
                            // Write 32 bit data value to DXR
                            MCBSP_write32(mhMcbsp,0x00000000);                  //make sure transciever LED turns off

                            // Wait for XRDY signal before writing data to DXR
                            while (!MCBSP_xrdy(mhMcbsp));
                            // Write 32 bit data value to DXR
                            MCBSP_write32(mhMcbsp,0x000000E0);                  //make sure transciever LED turns off

                            // Wait for XRDY signal before writing data to DXR
                            while (!MCBSP_xrdy(mhMcbsp));
                            // Write 32 bit data value to DXR
                            MCBSP_write32(mhMcbsp,0xC0000000);                  //make sure transciever LED turns off

                            // Wait for XRDY signal before writing data to DXR
                            while (!MCBSP_xrdy(mhMcbsp));
                            // Write 32 bit data value to DXR
                            MCBSP_write32(mhMcbsp, ErrCode);
```

```
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, transmit_wordMsb[0]);

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,0x00000000);          //give power supply a chance

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, transmit_wordMsb[1]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, transmit_wordMsb[2]);

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,0x00000000);          //give power supply a chance

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, transmit_wordLsb[0]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, transmit_wordLsb[1]);

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,0x00000000);          //give power supply a chance

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, transmit_wordLsb[2]);

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));     //make sure transciever LED turns off, and prevent
MCBSP_write32(mhMcbsp,0x00000000);  //interference from disrupting protocol,can make as many as
base has to do 50uS wait
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
MCBSP_write32(mhMcbsp,0x00000000);          //make sure transciever LED turns off
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
MCBSP_write32(mhMcbsp,0x00000000);          //make sure transciever LED turns off


//-------------------Keep DAC synced with ADC-----------------
if (tag == 1) {
        xmt1[0] = HdPhoneSampleR[0]; //for some reason DMA performs two transmissions per
audio sample
        xmt1[1] = HdPhoneSampleR[0]; //hence just repeat each sample
        xmt2[0] = HdPhoneSampleL[0];
        xmt2[1] = HdPhoneSampleL[0];

        xmt1[2] = HdPhoneSampleR[1];
        xmt1[3] = HdPhoneSampleR[1];
        xmt2[2] = HdPhoneSampleL[1];
        xmt2[3] = HdPhoneSampleL[1];

        xmt1[4] = HdPhoneSampleR[2];
        xmt1[5] = HdPhoneSampleR[2];
        xmt2[4] = HdPhoneSampleL[2];
        xmt2[5] = HdPhoneSampleL[2];}

if (tag == 0) {
        xmt1[0] = xmtoldR;   //for some reason DMA performs two transmissions per audio sample
        xmt1[1] = xmtoldR;   //hence just repeat each sample
```

```
                        xmt2[0] = xmtoldL;
                        xmt2[1] = xmtoldL;

                        xmt1[2] = xmtoldR;
                        xmt1[3] = xmtoldR;
                        xmt2[2] = xmtoldL;
                        xmt2[3] = xmtoldL;

                        xmt1[4] = xmtoldR;
                        xmt1[5] = xmtoldR;
                        xmt2[4] = xmtoldL;
                        xmt2[5] = xmtoldL;}

                DMA_RGETH(hDmaXmt2, DMACSR);
                DMA_RGETH(hDmaXmt1, DMACSR);

        //---------------------------------------------------------------

                if ((different_base == 1) && (correct_base_LOS < 100)) {correct_base_LOS++;}

                //reduce number of bits sent by a quarter the POTS is only 8 bits anyway and
                //the power supply noise is counterproductive anyway
                if ((MicSamplei[0] & 0x0008) != 0) {MicSamplei[0] |= 0x0010;}//half roundoff error
                if ((MicSamplei[1] & 0x0008) != 0) {MicSamplei[1] |= 0x0010;}//half roundoff error
                if ((MicSamplei[2] & 0x0008) != 0) {MicSamplei[2] |= 0x0010;}//half roundoff error
          MicSamplei[0] = MicSamplei[0] & 0xFFF0;
          MicSamplei[1] = MicSamplei[1] & 0xFFF0;
          MicSamplei[2] = MicSamplei[2] & 0xFFF0;

                ErrCode = 0;
                encodeIRword(MicSamplei[0], &transmit_wordLsb[0], &transmit_wordMsb[0]);
                encodeIRword(MicSamplei[1], &transmit_wordLsb[1], &transmit_wordMsb[1]);
                encodeIRword(MicSamplei[2], &transmit_wordLsb[2], &transmit_wordMsb[2]);

          crc8_codeTx = 0;
          for (i=0 ; i<3 ; i=i+1)                    // generate CRC checksum from transmitted data
                  {
                  tempTx = ((MicSamplei[i]) & 0xF000) >> 8;
                  indexTx = ((Uchar) tempTx) ^ crc8_codeTx;
                  crc8_codeTx = crc8_data[indexTx];
                  }

                encodeIRword(crc8_codeTx, &ErrCode, &dummy);

        }
        else {tag = 0;}         //if different base is present dont play audio

        asm(" NOP");
        asm(" NOP");
        *PCR = 0x0B00;
        asm(" NOP");
        asm(" NOP");
        *SPCR1 = 0x0001;                //take MCBSP reciever out of reset
        asm(" NOP");
        asm(" NOP");
        //asm("      BCLR XF ; clear xf pin");

   //IRQ_enable(IRQ_EVT_RINT0);
tag = 0;
}

/*************************************************************************************
 *-----------------------------Recieved IR transmission----------------------------
 *************************************************************************************/
void IRdata_recievedHWI(void)
{
        while (!MCBSP_xrdy(mhMcbsp));
        // Write 32 bit data value to DXR
        MCBSP_write16(mhMcbsp,0x0000);          //make sure transciever LED turns off

   while (!MCBSP_rrdy(mhMcbsp));
   dummy = MCBSP_read32(mhMcbsp);

        /* Wait for RRDY signal to read data from DRR */
   while (!MCBSP_rrdy(mhMcbsp));
   /* Read 32 bit value from DRR */
```

```
BaseIrCode = MCBSP_read32(mhMcbsp);
            /* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordLmsb[0] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordLlsb[0] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordRmsb[0] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordRlsb[0] = MCBSP_read32(mhMcbsp);

            /* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordLmsb[1] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordLlsb[1] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordRmsb[1] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordRlsb[1] = MCBSP_read32(mhMcbsp);

            /* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordLmsb[2] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordLlsb[2] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordRmsb[2] = MCBSP_read32(mhMcbsp);
/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
recieved_wordRlsb[2] = MCBSP_read32(mhMcbsp);

/* Wait for RRDY signal to read data from DRR */
while (!MCBSP_rrdy(mhMcbsp));
/* Read 32 bit value from DRR */
RcvCode = MCBSP_read32(mhMcbsp);

while (!MCBSP_rrdy(mhMcbsp));
dummy = MCBSP_read32(mhMcbsp);
while (!MCBSP_rrdy(mhMcbsp));
dummy = MCBSP_read32(mhMcbsp);

            asm(" NOP");
            asm(" NOP");      // Put MCBSP reciever into reset, NEED THIS HERE otherwise RINTO ISR
            *SPCR1 = 0x0000;  // preempts itself
            asm(" NOP");
            asm(" NOP");
            *PCR = 0x0A00;
            asm(" NOP");
            asm(" NOP");

            while (!MCBSP_xrdy(mhMcbsp));
            // Write 32 bit data value to DXR
            MCBSP_write16(mhMcbsp,0x0000);          //make sure transciever LED turns off

            HdPhoneSampleLold[2] = HdPhoneSampleL[2];
            HdPhoneSampleRold[2] = HdPhoneSampleR[2];

            asm("        BSET XF  ; set xf pin");
            decodeIRword(&HdPhoneSampleL[0], recieved_wordLlsb[0], recieved_wordLmsb[0]);  //decode trasmitted words
            decodeIRword(&HdPhoneSampleR[0], recieved_wordRlsb[0], recieved_wordRmsb[0]);  //into audio samples
            decodeIRword(&HdPhoneSampleL[1], recieved_wordLlsb[1], recieved_wordLmsb[1]);
            decodeIRword(&HdPhoneSampleR[1], recieved_wordRlsb[1], recieved_wordRmsb[1]);
            decodeIRword(&HdPhoneSampleL[2], recieved_wordLlsb[2], recieved_wordLmsb[2]);
            decodeIRword(&HdPhoneSampleR[2], recieved_wordRlsb[2], recieved_wordRmsb[2]);

            dummy = 0;
            decodeIRword(&CdChk, RcvCode, dummy);                //decode tranmitted CRC checksum
            decodeIRword(&TxIrCode, BaseIrCode, dummy);         //decode tranmitted CRC checksum

   crc8_codeRcv = 0;
   for (i=0 ; i<3 ; i++)                   // generate CRC checksum from transmitted data
               {
               tempRcv = ((HdPhoneSampleL[i]) & 0xF000) >> 8;
               indexRcv = ((Uchar) tempRcv) ^ crc8_codeRcv;
               crc8_codeRcv = crc8_data[indexRcv];

               tempRcv = ((HdPhoneSampleR[i]) & 0xF000) >> 8;
               indexRcv = ((Uchar) tempRcv) ^ crc8_codeRcv;
               crc8_codeRcv = crc8_data[indexRcv];
               }
            crc8_codeRcv = crc8_codeRcv & 0xFC;
            CdChk = CdChk & 0xFC;

            xmtoldR = xmt1[5];    //store previous correct and filtered audio sample values
            xmtoldL = xmt2[5];

            token = 0;
            if ((abs(HdPhoneSampleLold[2]))  - HdPhoneSampleL[0]> limit) {token = 1;}
            if ((abs(HdPhoneSampleRold[2]))  - HdPhoneSampleR[0]> limit) {token = 1;}
            if ((abs(HdPhoneSampleLold[2]))  - HdPhoneSampleL[1]> limit) {token = 1;}
            if ((abs(HdPhoneSampleRold[2]))  - HdPhoneSampleR[1]> limit) {token = 1;}
            if ((abs(HdPhoneSampleLold[2]))  - HdPhoneSampleL[2]> limit) {token = 1;}
            if ((abs(HdPhoneSampleRold[2]))  - HdPhoneSampleR[2]> limit) {token = 1;}

   //if limit is too low then get glitches every now and then even when there are no errors
            if ( (abs(HdPhoneSampleL[0]) < 20000)
                        && (abs(HdPhoneSampleR[0]) < 20000)
                        && (abs(HdPhoneSampleL[1]) < 20000)
                        && (abs(HdPhoneSampleR[1]) < 20000)
                        && (abs(HdPhoneSampleL[2]) < 20000)
                        && (abs(HdPhoneSampleR[2]) < 20000)) {token = 0;}

            if ((crc8_codeRcv != CdChk) || (token == 1))  {  //compare the generated and transmitted CRC checksums
                        HdPhoneSampleL[0] = xmtoldL;                      //and flatline if errors are detected
                        HdPhoneSampleR[0] = xmtoldR;
                        HdPhoneSampleL[1] = xmtoldL;
                        HdPhoneSampleR[1] = xmtoldR;
                        HdPhoneSampleL[2] = xmtoldL;
                        HdPhoneSampleR[2] = xmtoldR;
                        limit = 20000;                           //if an error occurred in the last transmission limit volume variation to
half peak
                        tag = 0;
                        }
            else {
                        limit = 32700;
                        tag = 1; }   //if no error occured in last transmission allow full volume range

            if (TxIrCode == Headset_code) {
               IRQ_disable(IRQ_EVT_RINT0);
                        correct_base_LOS = 0;
                        different_base = 0;}    //different base is in LOS
            else {different_base = 1;}
            asm("        BCLR XF  ; set xf pin");
}


/*********************************************************************************
 *                        ======= MAIN =======
 *********************************************************************************/
Void main()
{
```

```
            SPCR1 = (volatile ioport unsigned int*)0x2804;
            SPCR2 = (volatile ioport unsigned int*)0x2805;
            SRGR1 = (volatile ioport unsigned int*)0x280A;
            SRGR2 = (volatile ioport unsigned int*)0x280B;
            PCR = (volatile ioport unsigned int*)0x2812;

   PLL_config(&MyPLLConfig);
            EMIF_RSET(GBLCTL2, 0x9);  //make codec clock 3MHz
            /*
            mode = 1 means PLL enabled (non-bypass mode)
            mul = 5 means multiply by 5
            div0 = 0 means Divider0 divides by 1
            div1 = 3 means Divider1 divides by 4
            div2 = 3 means Divider2 divides by 4
            div3 = 3 means Divider3 divides by 4
            oscdiv = 1 means Oscillator Divider1 divides by 2
            */
            PLL_setFreq(1, 2, 0, 0, 0, 1, 0);

            // Open MCBSP Port 0, thiswill return a MCBSP handle that will be used in calls to other CSl functions.
            mhMcbsp = MCBSP_open(MCBSP_PORT0, MCBSP_OPEN_RESET);

            *SRGR1 = 0x0102; //SRGR1, SRGR2 do not get set by config, so do manually, are before cofig
            *SRGR2 = 0x200F; // because to change McBSP settings, port must be reset
            // Write configuration structure values to MCBSP control registers
            MCBSP_config(mhMcbsp, &ConfigLoopBack320);

             /* Start Sample Rate Generator and Frame Sync */
            MCBSP_start(mhMcbsp, MCBSP_SRGR_START | MCBSP_SRGR_FRAMESYNC,0x300);

            /* Enable MCBSP transmit and receive */
            MCBSP_start(mhMcbsp, MCBSP_RCV_START | MCBSP_XMIT_START, 0x200);

   // Open MCBSP Port 1, this will return a MCBSP handle that will be used in calls to other CSl functions.
            C55XX_CONTROLHANDLE_hMcbsp = MCBSP_open(MCBSP_PORT1, MCBSP_OPEN_RESET);

            // Write configuration structure values to MCBSP control registers
            MCBSP_config(C55XX_CONTROLHANDLE_hMcbsp, &ConfigMcbsp1);

            // Open MCBSP Port 2, this will return a MCBSP handle that will be used in calls to other CSl functions.
            C55XX_DMA_MCBSP_hMcbsp = MCBSP_open(MCBSP_PORT2, MCBSP_OPEN_RESET);

            // Write configuration structure values to MCBSP control registers
            MCBSP_config(C55XX_DMA_MCBSP_hMcbsp, &ConfigMcbsp2);

   // Start McBSP1 as the codec control channel
   MCBSP_start(C55XX_CONTROLHANDLE_hMcbsp,      MCBSP_XMIT_START      |      MCBSP_SRGR_START      |
MCBSP_SRGR_FRAMESYNC, 100);

   // Reset the AIC23
   my_AIC23_rset(0, my_AIC23_RESET, 0);

   // Configure the rest of the AIC23 registers
   my_AIC23_config(0, &config);

            // Clear any garbage from the codec data port
   if (MCBSP_rrdy(C55XX_DMA_MCBSP_hMcbsp))
      MCBSP_read16(C55XX_DMA_MCBSP_hMcbsp);

   // Start McBSP2 as the codec data channel
   MCBSP_start(C55XX_DMA_MCBSP_hMcbsp, MCBSP_XMIT_START | MCBSP_RCV_START |
            MCBSP_SRGR_START | MCBSP_SRGR_FRAMESYNC, 0x300);

   /* Open DMA Channel 0 */
   hDmaXmt2 = DMA_open(DMA_CHA0, 0);
   hDmaXmt1 = DMA_open(DMA_CHA1, 0);
   hDmaRcv = DMA_open(DMA_CHA4, 0);

   /* By default, the TMS320C55xx compiler assigns all data symbols word */
   /* addresses. The DMA however, expects all addresses to be byte       */
   /* addresses. Therefore, we must shift the address by 2 in order to   */
   /* change the word address to a byte address for the DMA transfer.    */
   srcAddrHi = (Uint16)(((Uint32)(MCBSP_ADDR(DRR12))) >> 15) & 0xFFFFu;
   srcAddrLo = (Uint16)(((Uint32)(MCBSP_ADDR(DRR12))) << 1) & 0xFFFFu;
   dstAddrHi = (Uint16)(((Uint32)(&MicSample[0])) >> 15) & 0xFFFFu;
   dstAddrLo = (Uint16)(((Uint32)(&MicSample[0])) << 1) & 0xFFFFu;
   dmaRcvConfig.dmacssal = (DMA_AdrPtr)srcAddrLo;
   dmaRcvConfig.dmacssau = srcAddrHi;
   dmaRcvConfig.dmacdsal = (DMA_AdrPtr)dstAddrLo;
   dmaRcvConfig.dmacdsau = dstAddrHi;

            srcAddrHi = (Uint16)(((Uint32)(&xmt1[0])) >> 15) & 0xFFFFu;
   srcAddrLo = (Uint16)(((Uint32)(&xmt1[0])) << 1) & 0xFFFFu;
   dstAddrHi = (Uint16)(((Uint32)(MCBSP_ADDR(DXR12))) >> 15) & 0xFFFFu;
   dstAddrLo = (Uint16)(((Uint32)(MCBSP_ADDR(DXR12))) << 1) & 0xFFFFu;
   dmaXmtConfig1.dmacssal = (DMA_AdrPtr)srcAddrLo;
   dmaXmtConfig1.dmacssau = srcAddrHi;
   dmaXmtConfig1.dmacdsal = (DMA_AdrPtr)dstAddrLo;
   dmaXmtConfig1.dmacdsau = dstAddrHi;

            srcAddrHi = (Uint16)(((Uint32)(&xmt2[0])) >> 15) & 0xFFFFu;
   srcAddrLo = (Uint16)(((Uint32)(&xmt2[0])) << 1) & 0xFFFFu;
   dstAddrHi = (Uint16)(((Uint32)(MCBSP_ADDR(DXR22))) >> 15) & 0xFFFFu;
   dstAddrLo = (Uint16)(((Uint32)(MCBSP_ADDR(DXR22))) << 1) & 0xFFFFu;
   dmaXmtConfig2.dmacssal = (DMA_AdrPtr)srcAddrLo;
   dmaXmtConfig2.dmacssau = srcAddrHi;
   dmaXmtConfig2.dmacdsal = (DMA_AdrPtr)dstAddrLo;
   dmaXmtConfig2.dmacdsau = dstAddrHi;

            DmaRcvEventId = DMA_getEventId(hDmaRcv);
            IRQ_globalDisable();
            IRQ_clear(DmaRcvEventId);
            IRQ_enable(DmaRcvEventId);    //enable in intm register

   /* Write configuration structure values to DMA control registers */
   DMA_config(hDmaXmt2, &dmaXmtConfig2);
   DMA_config(hDmaXmt1, &dmaXmtConfig1);
   DMA_config(hDmaRcv, &dmaRcvConfig);

   /* Enable DMA channel to begin transfer */
   DMA_start(hDmaRcv);
   DMA_start(hDmaXmt2);
   DMA_start(hDmaXmt1);

   IRQ_enable(IRQ_EVT_RINT0);              //enable in intm register

            while((!my_AIC23_write32(0x00000000)) && (delay < 10)) {delay++;}   //must clear any garbage from McBSP
port for DMA to work
            delay = 0;
            while((!my_AIC23_read16(&MicSample[0])) && (delay < 1000)) {delay++;}    //must clear any garbage from
McBSP port for DMA to work
            delay = 0;

            if (delay == 5) {
                    IRdata_recievedHWI();
                    IRdata_transmitHWI();
                    }

            asm(" NOP");
            asm(" NOP");      // initially, Put MCBSP reciever into reset, and prevent noise interference
            *SPCR1 = 0x0000;  // from disrupting the protocol
            asm(" NOP");
            asm(" NOP");
            *PCR = 0x0A00;
            asm(" NOP");
            asm(" NOP");

            DMA_RGETH(hDmaXmt2, DMACSR);
            DMA_RGETH(hDmaXmt1, DMACSR);
            DMA_RGETH(hDmaRcv, DMACSR);

   IDL_run();                                    //fall back to kernal and real time schedular.
}
```

II.b. Base Station Source Code

```
/*5502 IR Base Station Code Code
2005-08-25: Cleaned up code
Andrew Pasquale, Curtin University of Technology*/

#define CHIP_5502 1

#include <log.h>
#include <clk.h>
#include <hwi.h>
#include <csl_mcbsp.h>
#include <csl_irq.h>
#include <csl_pll.h>
#include <csl_emif.h>
#include <csl_chip.h>
#include <intrindefs.h>
#include <csl_dma.h>
#include <csl_gpt.h>
#include "clktestcfg.h"

void IRdata_recievedHWI(void);

MCBSP_Handle mhMcbsp, C55XX_CONTROLHANDLE_hMcbsp, C55XX_DMA_MCBSP_hMcbsp;

#define N       6

/* Define transmit and receive buffers */
#pragma DATA_SECTION(AudioInSampleR,"dmaMem")
Int16 AudioInSampleR[40];
#pragma DATA_SECTION(AudioInSampleL,"dmaMem")
Int16 AudioInSampleL[40];

#pragma DATA_SECTION(BaseOutSample,"dmaMem")
Int16 BaseOutSample[40];

#define my_AIC23_RESET       15
#define my_AIC23_NUMREGS     10
typedef int my_AIC23_CodecHandle;
my_AIC23_CodecHandle hCodec;

typedef struct my_AIC23_Config {              /* Parameter Structure for the AIC23 Codec */
   int regs[my_AIC23_NUMREGS];
} my_AIC23_Config;

#define my_AIC23_DEFAULTCONFIG { \
   0x0017, /* Set-Up Reg 0       Left line input channel volume control */  \
         /* LRS     0          simultaneous left/right volume: disabled */\
         /* LIM     0          left line input mute: disabled */         \
         /* XX      00         reserved */                               \
         /* LIV     10111      left line input volume: 0 dB */           \
                                                                        \
   0x0017, /* Set-Up Reg 1       Right line input channel volume control */ \
         /* RLS     0          simultaneous right/left volume: disabled */\
         /* RIM     0          right line input mute: disabled */        \
         /* XX      00         reserved */                               \
         /* RIV     10111      right line input volume: 0 dB */          \
                                                                        \
   0x01f9, /* Set-Up Reg 2       Left channel headphone volume control */   \
         /* LRS     1          simultaneous left/right volume: enabled */ \
         /* LZC     1          left channel zero-cross detect: enabled */ \
         /* LHV     1111001    left headphone volume: 0 dB */            \
                                                                        \
   0x01f9, /* Set-Up Reg 3       Right channel headphone volume control */  \
         /* RLS     1          simultaneous right/left volume: enabled */ \
         /* RZC     1          right channel zero-cross detect: enabled */\
         /* RHV     1111001    right headphone volume: 0 dB */           \
                                                                        \
   0x0011, /* Set-Up Reg 4       Analog audio path control */              \
         /* X       0          reserved */                               \
         /* STA     00         sidetone attenuation: -6 dB */            \
         /* STE     0          sidetone: disabled */                     \
         /* DAC     1         DAC: selected */                        \
         /* BYP     0         bypass: off */                          \
         /* INSEL   0         input select for ADC: line */           \
         /* MICM    0         microphone mute: disabled */            \
         /* MICB    1         microphone boost: enabled */            \
                                                                        \
   0x0000, /* Set-Up Reg 5       Digital audio path control */      \
         /* XXXXX   00000     reserved */                             \
         /* DACM    0         DAC soft mute: disabled */              \
         /* DEEMP   00        deemphasis control: disabled */         \
         /* ADCHP   0         ADC high-pass filter: disabled */       \
                                                                        \
   0x0000, /* Set-Up Reg 6       Power down control */              \
         /* X       0         reserved */                             \
         /* OFF     0         device power: on (i.e. not off) */      \
         /* CLK     0         clock: on */                            \
         /* OSC     0         oscillator: on */                       \
         /* OUT     0         outputs: on */                          \
         /* DAC     0         DAC: on */                              \
         /* ADC     0         ADC: on */                              \
         /* MIC     0         microphone: on */                       \
         /* LINE    0         line input: on */                       \
                                                                        \
   0x0043, /* Set-Up Reg 7       Digital audio interface format */  \
         /* XX      00        reserved */                             \
         /* MS      1         master/slave mode: master */            \
         /* LRSWAP  0         DAC left/right swap: disabled */        \
         /* LRP     0         DAC lrp: MSB on 1st BCLK */             \
         /* IWL     00        input bit length: 16 bit */             \
         /* FOR     11        data format: DSP format */              \
                                                                        \
   0x0081, /* Set-Up Reg 8       Sample rate control */             \
         /* X       0         reserved */                             \
         /* CLKOUT  1         clock output divider: 2 (MCLK/2) */     \
         /* CLKIN   0         clock input divider: 2 (MCLK/2) */      \
         /* SR,BOSR 00000     sampling rate: ADC  48 kHz DAC  48 kHz */  \
         /* USB/N   1         clock mode select (USB/normal): USB */  \
                                                                        \
   0x0001  /* Set-Up Reg 9       Digital interface activation */    \
         /* XX..X   00000000   reserved */                            \
         /* ACT     1         active */                               \
}

/* Codec configuration settings */
my_AIC23_Config config = {
   0x001F,  /* 0 DSK5510_AIC23_LEFTINVOL  Left line input channel volume */ \
   0x001F,  /* 1 DSK5510_AIC23_RIGHTINVOL Right line input channel volume */\
   0x016F,  /* 2 DSK5510_AIC23_LEFTHPVOL  Left channel headphone volume */  \
   0x016F,  /* 3 DSK5510_AIC23_RIGHTHPVOL Right channel headphone volume */ \
   0x0012,  /* 4 DSK5510_AIC23_ANAPATH    Analog audio path control, line input */     \
   0x0001,  /* 5 DSK5510_AIC23_DIGPATH    Digital audio path control, with high pass filter */     \
   0x0062,  /* 6 DSK5510_AIC23_POWERDOWN  Power down control */            \
   0x0043,  /* 7 DSK5510_AIC23_DIGIF      Digital audio interface format */ \
   0x0084,  /* 8 DSK5510_AIC23_SAMPLERATE Sample rate control, ADC at 48KHz, DAC at 8KHz */            \
   0x0001   /* 9 DSK5510_AIC23_DIGACT     Digital interface activation */   \
};

//Pre-initialized configuration structure for PLL
PLL_Config MyPLLConfig =
{
         PLL_PLLCSR_RMK(
                 PLL_PLLCSR_PLLRST_RESET_RELEASED,
                 PLL_PLLCSR_OSCPWRDN_OSC_ON,
                 PLL_PLLCSR_PLLPWRDN_PLL_ON,
                 PLL_PLLCSR_PLLEN_DEFAULT
         ),
         PLL_PLLM_PLLM_OF(10),
         PLL_PLLDIV0_RMK(
                 PLL_PLLDIV0_D0EN_ENABLED,
                 PLL_PLLDIV0_PLLDIV0_OF(0)
         ),
         PLL_PLLDIV1_RMK(
                 PLL_PLLDIV1_D1EN_ENABLED,
                 PLL_PLLDIV1_PLLDIV1_OF(0)
         ),
         PLL_PLLDIV2_RMK(
                        PLL_PLLDIV2_D2EN_ENABLED,
                        PLL_PLLDIV2_PLLDIV2_OF(0)
                ),
                PLL_PLLDIV3_RMK(
                        PLL_PLLDIV3_D3EN_ENABLED,
                        PLL_PLLDIV3_PLLDIV3_OF(0)
                ),
                PLL_OSCDIV1_RMK(
                        PLL_OSCDIV1_OD1EN_DISABLED,
                        PLL_OSCDIV1_OSCDIV1_OF(0)
                ),
                PLL_WKEN_RMK(
                        PLL_WKEN_WKEN4_ENABLED,
                        PLL_WKEN_WKEN3_ENABLED,
                        PLL_WKEN_WKEN2_ENABLED,
                        PLL_WKEN_WKEN1_ENABLED,
                        PLL_WKEN_WKEN0_ENABLED
                ),
                PLL_CLKMD_RMK(
                        PLL_CLKMD_CLKMD0_OSCOUT
                ),
                PLL_CLKOUTSR_RMK(
                        PLL_CLKOUTSR_CLKOSEL_SYSCLK1,
                        PLL_CLKOUTSR_CLKOUTDIS_ENABLED
                )
        };

        GPT_Config MyConfig = {
         0x0003, //gptemu   Emulation management register
         0x0000, //gptgpint GPIO interrupt control register
         0x0000, //gptgpen  GPIO enable register
         0x0000, //gptgpdir GPIO direction register
         0x0000, //gptgpdat GPIO data register
         0x40EC, //gptprd1  Timer period register 1, if CPU at 96MHz, to get 1301Hz need to divide
         0x0002, //gptprd2  Timer period register 2, by 1203Dh
         0x40EC, //gptprd3  Timer period register 3
         0x0002, //gptprd4  Timer period register 4
         0x0080, //gptctl1  Timer control register 1
         0x0080, //gptctl2  Timer control register 2
         0x0007 //gptgctl1 Global timer control register
        };

        /* Create a MCBSP configuration structure */
          MCBSP_Config ConfigLoopBack320= {
          MCBSP_SPCR1_RMK(
            MCBSP_SPCR1_DLB_OFF,                  /* DLB    = 0 */
            MCBSP_SPCR1_RJUST_RZF,                 /* RJUST  = 0 */
            MCBSP_SPCR1_CLKSTP_DISABLE,            /* CLKSTP = 0 */
            MCBSP_SPCR1_DXENA_NA,                 /* DXENA  = 0 */
            MCBSP_SPCR1_ABIS_DISABLE,              /* ABIS   = 0 */
            MCBSP_SPCR1_RINTM_RRDY,                /* RINTM  = 0 */
            0,                            /* RSYNCER = 0 */
            MCBSP_SPCR1_RRST_DISABLE               /* RRST   = 0 */
           ),
            MCBSP_SPCR2_RMK(
            MCBSP_SPCR2_FREE_NO,                  /* FREE   = 0 */
            MCBSP_SPCR2_SOFT_NO,                  /* SOFT   = 0 */
            MCBSP_SPCR2_FRST_FSG,                 /* FRST   = 1 */
            MCBSP_SPCR2_GRST_CLKG,                /* GRST   = 1 */
            MCBSP_SPCR2_XINTM_XRDY,                /* XINTM  = 0 */
            0,                            /* XSYNCER = N/A */
            MCBSP_SPCR2_XRST_DISABLE               /* XRST   = 0 */
           ),
          MCBSP_RCR1_RMK(
          MCBSP_RCR1_RFRLEN1_OF(12),               /* RFRLEN1 = 3 */
          MCBSP_RCR1_RWDLEN1_32BIT                 /* RWDLEN1 = 5 */
          ),
         MCBSP_RCR2_RMK(
            MCBSP_RCR2_RPHASE_SINGLE,              /* RPHASE  = 0 */
            MCBSP_RCR2_RFRLEN2_OF(0),             /* RFRLEN2 = 0 */
            MCBSP_RCR2_RWDLEN2_8BIT,               /* RWDLEN2 = 0 */
            MCBSP_RCR2_RCOMPAND_MSB,               /* RCOMPAND = 0 */
            MCBSP_RCR2_RFIG_NO,                  /* RFIG    = 1 */
            MCBSP_RCR2_RDATDLY_1BIT                /* RDATDLY = 1 */
            ),
           MCBSP_XCR1_RMK(
            MCBSP_XCR1_XFRLEN1_OF(12),              /* XFRLEN1 = 1 */
            MCBSP_XCR1_XWDLEN1_32BIT               /* XWDLEN1 = 5 */

          ),
         MCBSP_XCR2_RMK(
            MCBSP_XCR2_XPHASE_SINGLE,              /* XPHASE  = 0 */
            MCBSP_XCR2_XFRLEN2_OF(0),             /* XFRLEN2 = 0 */
            MCBSP_XCR2_XWDLEN2_8BIT,              /* XWDLEN2 = 0 */
            MCBSP_XCR2_XCOMPAND_MSB,               /* XCOMPAND = 0 */
            MCBSP_XCR2_XFIG_NO,                  /* XFIG    = 1 */
            MCBSP_XCR2_XDATDLY_0BIT               /* XDATDLY = 0 */
           ),
          MCBSP_SRGR1_RMK(
           MCBSP_SRGR1_FWID_OF(1),               /* FWID    = 1 */
           MCBSP_SRGR1_CLKGDV_OF(11)              /* CLKGDV  = 2 */
          ),
         MCBSP_SRGR2_RMK(
            MCBSP_SRGR2_GSYNC_FREE,               /* FREE    = 1 */
            MCBSP_SRGR2_CLKSP_RISING,             /* CLKSP   = 0 */
            MCBSP_SRGR2_CLKSM_INTERNAL,           /* CLKSM   = 1 */
            MCBSP_SRGR2_FSGM_DXR2XSR,             /* FSGM    = 0 */
            MCBSP_SRGR2_FPER_OF(15)               /* FPER    = 15 */
         ),
         MCBSP_MCR1_DEFAULT,
         MCBSP_MCR2_DEFAULT,
         MCBSP_PCR_RMK(
           MCBSP_PCR_XIOEN_SP,                   /* XIOEN    = 0   */
           MCBSP_PCR_RIOEN_SP,                   /* RIOEN    = 0   */
           MCBSP_PCR_FSXM_INTERNAL,               /* FSXM     = 1   */
           MCBSP_PCR_FSRM_EXTERNAL,               /* FSRM     = 0   */
           MCBSP_PCR_CLKXM_OUTPUT,                /* CLKXM    = 1   */
           MCBSP_PCR_CLKRM_OUTPUT,                /* CLKRM    = 1   */
           0,                                 /* DXSTAT = N/A   */
           MCBSP_PCR_SCLKME_NO,                  /* SCLKME   = 0   */
           MCBSP_PCR_FSXP_ACTIVEHIGH,             /* FSXP     = 0   */
           MCBSP_PCR_FSRP_ACTIVEHIGH,             /* FSRP     = 0   */
           MCBSP_PCR_CLKXP_RISING,                /* CLKXP    = 0   */
           MCBSP_PCR_CLKRP_RISING                /* CLKRP    = 1   */
         ),
         MCBSP_RCERA_DEFAULT,
         MCBSP_RCERB_DEFAULT,
         MCBSP_RCERC_DEFAULT,
         MCBSP_RCERD_DEFAULT,
         MCBSP_RCERE_DEFAULT,
         MCBSP_RCERF_DEFAULT,
         MCBSP_RCERG_DEFAULT,
         MCBSP_RCERH_DEFAULT,
         MCBSP_XCERA_DEFAULT,
         MCBSP_XCERB_DEFAULT,
         MCBSP_XCERC_DEFAULT,
         MCBSP_XCERD_DEFAULT,
         MCBSP_XCERE_DEFAULT,
         MCBSP_XCERF_DEFAULT,
         MCBSP_XCERG_DEFAULT,
         MCBSP_XCERH_DEFAULT
        };

        MCBSP_Config ConfigMcbsp1 = {
        0x1000, /* spcr1 0001 0000 0000 0000, SPI (clock stop) mode enabled*/
        0x0100, /* spcr2 */
        0x0000, /* rcr1 */
        0x0000, /* rcr2 */
        0x0040, /* xcr1 */
        0x0002, /* xcr2 */
        0x000C, /* srgr1 0000 0000 0110 0011*/
        0x2013, /* srgr2 */
        0x0000, /* mcr1 */
        0x0000, /* mcr2 */
        0x0A0A, /* pcr, 0001 1010 0000 1010 */
        0xFFFF, /* rcera */
        0xFFFF, /* rcerb */
        0xFFFF, /* rcerc */
        0xFFFF, /* rcerd */
        0xFFFF, /* rcere */
        0xFFFF, /* rcerf */
        0xFFFF, /* rcerg */
```

```
0xFFFF, /* rcerh */
0xFFFF, /* xcera */
0xFFFF, /* xcerb */
0xFFFF, /* xcerc */
0xFFFF, /* xcerd */
0xFFFF, /* xcere */
0xFFFF, /* xcerf */
0xFFFF, /* xcerg */
0xFFFF /* xcerh */
};

MCBSP_Config ConfigMcbsp2 = {
0x0000, /* spcr1 */
0x0100, /* spcr2 */
0x00A0, /* rcr1 */
0x0001, /* rcr2 ,make rdatdly = 1, check new headset board, if 0 works since overflow can occur if user speaks loud*/
0x0140, /* xcr1, 32bit word for transmit */
0x0000, /* xcr2 */
0x0000, /* srgr1 */
0x2000, /* srgr2 */
0x0000, /* mcr1 */
0x0000, /* mcr2 */
0x0182, /* pcr, make clock polarity negative edge triggered, solves mic overflow problem */
0xFFFF, /* rcera */
0xFFFF, /* rcerb */
0xFFFF, /* rcerc */
0xFFFF, /* rcerd */
0xFFFF, /* rcere */
0xFFFF, /* rcerf */
0xFFFF, /* rcerg */
0xFFFF, /* rcerh */
0xFFFF, /* xcera */
0xFFFF, /* xcerb */
0xFFFF, /* xcerc */
0xFFFF, /* xcerd */
0xFFFF, /* xcere */
0xFFFF, /* xcerf */
0xFFFF, /* xcerg */
0xFFFF /* xcerh */
};


/* Create DMA Transmit Side Configuration */
DMA_Config  dmaRcvConfig1 = {
  DMA_DMACSDP_RMK(
    DMA_DMACSDP_DSTBEN_NOBURST,
    DMA_DMACSDP_DSTPACK_OFF,
    DMA_DMACSDP_DST_DARAMPORT1,
    DMA_DMACSDP_SRCBEN_NOBURST,
    DMA_DMACSDP_SRCPACK_OFF,
    DMA_DMACSDP_SRC_PERIPH,
    DMA_DMACSDP_DATATYPE_16BIT
  ),                                  /* DMACSDP  */
  DMA_DMACCR_RMK(
    DMA_DMACCR_DSTAMODE_POSTINC,
    DMA_DMACCR_SRCAMODE_CONST,
    DMA_DMACCR_ENDPROG_OFF,
    DMA_DMACCR_WP_ENABLE,
    DMA_DMACCR_REPEAT_ALWAYS,
    DMA_DMACCR_AUTOINIT_ON,
    DMA_DMACCR_EN_STOP,
    DMA_DMACCR_PRIO_HI,
    DMA_DMACCR_FS_DISABLE,
    DMA_DMACCR_SYNC_REVT2
  ),                                  /* DMACCR   */
  DMA_DMACICR_RMK(
    DMA_DMACICR_BLOCKIE_OFF,
    DMA_DMACICR_LASTIE_OFF,
    DMA_DMACICR_FRAMEIE_OFF,
    DMA_DMACICR_FIRSTHALFIE_OFF,
    DMA_DMACICR_DROPIE_OFF,
    DMA_DMACICR_TIMEOUTIE_OFF
  ),                                  /* DMACICR  */
    (DMA_AdrPtr)(MCBSP_ADDR(DRR12)),                     /* DMACSSAL */
    0,                                /* DMACSSAU */
    (DMA_AdrPtr)&AudioInSampleR[0],                                /* DMACDSAL */
    0,                              /* DMACDSAU */
    1,                              /* DMACEN   */
    20,                              /* DMACFN   */
    0,                                        /* DMACSFI  */
    0,                                                    /* DMACSEI  */
    0,                                        /* DMACDFI  */
    0                                         /* DMACDEI  */
};

DMA_Config  dmaRcvConfig2 = {
  DMA_DMACSDP_RMK(
    DMA_DMACSDP_DSTBEN_NOBURST,
    DMA_DMACSDP_DSTPACK_OFF,
    DMA_DMACSDP_DST_DARAMPORT1,
    DMA_DMACSDP_SRCBEN_NOBURST,
    DMA_DMACSDP_SRCPACK_OFF,
    DMA_DMACSDP_SRC_PERIPH,
    DMA_DMACSDP_DATATYPE_16BIT
  ),                               /* DMACSDP  */
  DMA_DMACCR_RMK(
    DMA_DMACCR_DSTAMODE_POSTINC,
    DMA_DMACCR_SRCAMODE_CONST,
    DMA_DMACCR_ENDPROG_OFF,
    DMA_DMACCR_WP_ENABLE,
    DMA_DMACCR_REPEAT_ALWAYS,
    DMA_DMACCR_AUTOINIT_ON,
    DMA_DMACCR_EN_STOP,
    DMA_DMACCR_PRIO_HI,
    DMA_DMACCR_FS_DISABLE,
    DMA_DMACCR_SYNC_REVT2
  ),                               /* DMACCR   */
  DMA_DMACICR_RMK(
    DMA_DMACICR_BLOCKIE_OFF,
    DMA_DMACICR_LASTIE_OFF,
    DMA_DMACICR_FRAMEIE_OFF,
    DMA_DMACICR_FIRSTHALFIE_OFF,
    DMA_DMACICR_DROPIE_OFF,
    DMA_DMACICR_TIMEOUTIE_OFF
  ),                               /* DMACICR  */
    (DMA_AdrPtr)(MCBSP_ADDR(DRR22)),                  /* DMACSSAL */
    0,                              /* DMACSSAU */
    (DMA_AdrPtr)&AudioInSampleL[0],                            /* DMACDSAL */
    0,                              /* DMACDSAU */
    1,                              /* DMACEN   */
    20,                              /* DMACFN   */
    0,                                        /* DMACSFI  */
    0,                                                    /* DMACSEI  */
    0,                                        /* DMACDFI  */
    0                                         /* DMACDEI  */
};


DMA_Config  dmaXmtConfig = {
  DMA_DMACSDP_RMK(
    DMA_DMACSDP_DSTBEN_NOBURST,
    DMA_DMACSDP_DSTPACK_OFF,
    DMA_DMACSDP_DST_PERIPH,
    DMA_DMACSDP_SRCBEN_NOBURST,
    DMA_DMACSDP_SRCPACK_OFF,
    DMA_DMACSDP_SRC_DARAMPORT0,
    DMA_DMACSDP_DATATYPE_16BIT
  ),                               /* DMACSDP  */
  DMA_DMACCR_RMK(
    DMA_DMACCR_DSTAMODE_CONST,
    DMA_DMACCR_SRCAMODE_POSTINC,
    DMA_DMACCR_ENDPROG_OFF,
    DMA_DMACCR_WP_DEFAULT,
    DMA_DMACCR_REPEAT_ALWAYS,
    DMA_DMACCR_AUTOINIT_ON,
    DMA_DMACCR_EN_STOP,
    DMA_DMACCR_PRIO_HI,
    DMA_DMACCR_FS_DISABLE,
    DMA_DMACCR_SYNC_XEVT2
  ),                               /* DMACCR   */
  DMA_DMACICR_RMK(
    DMA_DMACICR_BLOCKIE_OFF,
    DMA_DMACICR_LASTIE_OFF,
    DMA_DMACICR_FRAMEIE_OFF,
    DMA_DMACICR_FIRSTHALFIE_OFF,
    DMA_DMACICR_DROPIE_OFF,
    DMA_DMACICR_TIMEOUTIE_OFF
  ),                              /* DMACICR  */
    (DMA_AdrPtr)&BaseOutSample[0],                  /* DMACSSAL */
    0,                            /* DMACSSAU */
    (DMA_AdrPtr)(MCBSP_ADDR(DXR12)),       /* DMACDSAL */
    0,                            /* DMACDSAU */
    1,                            /* DMACEN   */
    6,                            /* DMACFN   */
    1,                                      /* DMACSFI  */
    1,                                                /* DMACSEI  */
    0,                                      /* DMACDFI  */
    0                                       /* DMACDEI  */
};

#define Headset_code              0x00110100              //headset number is 0x34, is arbitrary

Int32 dummy, HdPhone;
Int16         BaseOutSampleOld[N], tempTx, tempRcv, CdChk, AudioInSampleLj[25], AudioInSampleRj[25],
                        LPAudioInSampleL[N], LPAudioInSampleR[N], BaseOutSamplei[7],
                        AudioInSampleLi[25], AudioInSampleRi[25], AudioInSampleLold, AudioInSampleRold;
Uint32       transmit_wordLmsb[N], transmit_wordRmsb[N], transmit_wordLlsb[N], transmit_wordRlsb[N],
                        delay = 0, recieved_wordMsb[N], recieved_wordLsb[N], ErrCode, RcvCode;
MCBSP_Handle mhMcbsp;
int i=0, j=0, token = 0, syncL = 0, syncR = 0;
Int16 *A, *B, *C, *D, z[4];
Uchar crc8_codeTx, indexTx, crc8_codeRcv, indexRcv;
Uint16 limit = 32000;

/* Define a DMA_Handle object to be used with DMA_open function */
DMA_Handle hDmaRcv1, hDmaRcv2, hDmaXmt, hDmaXmt2;
Uint16 srcAddrHi, srcAddrLo;
Uint16 dstAddrHi, dstAddrLo;
Uint16 xmtEventId, rcvEventId;
Uint16 old_intm, dmastat0, dmastat1, dmastat2;
Int16 ElmtR, ElmtL, ElmtRp2, ElmtLp2, ElmtRm18, ElmtLm18,AudioInSampleLaddr, AudioInSampleRaddr;
Int16 BaseOutSampleAddr, ElmtB, ElmtBm9, ElmtBp6;

/* Internal codec state used to simulate read/write functionality */
static my_AIC23_Config codecstate = my_AIC23_DEFAULTCONFIG;
Uint32 regval, regnum;
Uint16 sample;                                    //limit of audio volume variation

GPT_Handle hGpt;
GPT_Config MyConfig;
Uint32 *tim12,*tim34, PeriodValue;
Uint16 t1, t2;

Uchar crc8_data[] = { //CRC checksum lookup table
0x00, 0x5e, 0xbc, 0xe2, 0x61, 0x3f, 0xdd, 0x83,
0xc2, 0x9c, 0x7e, 0x20, 0xa3, 0xfd, 0x1f, 0x41,
0x9d, 0xc3, 0x21, 0x7f, 0xfc, 0xa2, 0x40, 0x1e,
0x5f, 0x01, 0xe3, 0xbd, 0x3e, 0x60, 0x82, 0xdc,
0x23, 0x7d, 0x9f, 0xc1, 0x42, 0x1c, 0xfe, 0xa0,
0xe1, 0xbf, 0x5d, 0x03, 0x80, 0xde, 0x3c, 0x62,
0xbe, 0xe0, 0x02, 0x5c, 0xdf, 0x81, 0x63, 0x3d,
0x7c, 0x22, 0xc0, 0x9e, 0x1d, 0x43, 0xa1, 0xff,
0x46, 0x18, 0xfa, 0xa4, 0x27, 0x79, 0x9b, 0xc5,
0x84, 0xda, 0x38, 0x66, 0xe5, 0xbb, 0x59, 0x07,
0xdb, 0x85, 0x67, 0x39, 0xba, 0xe4, 0x06, 0x58,
0x19, 0x47, 0xa5, 0xfb, 0x78, 0x26, 0xc4, 0x9a,
0x65, 0x3b, 0xd9, 0x87, 0x04, 0x5a, 0xb8, 0xe6,
0xa7, 0xf9, 0x1b, 0x45, 0xc6, 0x98, 0x7a, 0x24,
0xf8, 0xa6, 0x44, 0x1a, 0x99, 0xc7, 0x25, 0x7b,
0x3a, 0x64, 0x86, 0xd8, 0x5b, 0x05, 0xe7, 0xb9,
0x8c, 0xd2, 0x30, 0x6e, 0xed, 0xb3, 0x51, 0x0f,
0x4e, 0x10, 0xf2, 0xac, 0x2f, 0x71, 0x93, 0xcd,
0x11, 0x4f, 0xad, 0xf3, 0x70, 0x2e, 0xcc, 0x92,
0xd3, 0x8d, 0x6f, 0x31, 0xb2, 0xec, 0x0e, 0x50,
0xaf, 0xf1, 0x13, 0x4d, 0xce, 0x90, 0x72, 0x2c,
0x6d, 0x33, 0xd1, 0x8f, 0x0c, 0x52, 0xb0, 0xee,
0x32, 0x6c, 0x8e, 0xd0, 0x53, 0x0d, 0xef, 0xb1,
0xf0, 0xae, 0x4c, 0x12, 0x91, 0xcf, 0x2d, 0x73,
0xca, 0x94, 0x76, 0x28, 0xab, 0xf5, 0x17, 0x49,
0x08, 0x56, 0xb4, 0xea, 0x69, 0x37, 0xd5, 0x8b,
0x57, 0x09, 0xeb, 0xb5, 0x36, 0x68, 0x8a, 0xd4,
0x95, 0xcb, 0x29, 0x77, 0xf4, 0xaa, 0x48, 0x16,
0xe9, 0xb7, 0x55, 0x0b, 0x88, 0xd6, 0x34, 0x6a,
0x2b, 0x75, 0x97, 0xc9, 0x4a, 0x14, 0xf6, 0xa8,
0x74, 0x2a, 0xc8, 0x96, 0x15, 0x4b, 0xa9, 0xf7,
0xb6, 0xe8, 0x0a, 0x54, 0xd7, 0x89, 0x6b, 0x35,
};

#define ORDER          601

Int16 delayBufferL[ORDER+2]={0};    // FirDec Delay Buffers for DSPLIB routine
Int16 delayBufferR[ORDER+2]={0};
/*Create a pointer to hold the addresses of the buffers start
 * The FIR routine requires pointer indirection*/
Int16 *delayPtrL = &(delayBufferL[0]);
Int16 *delayPtrR = &(delayBufferR[0]);

Int16 COEFFS[ORDER] = { //Order 601, Fs = 46.875KHz, Ps = 3750, St = 3906
                -20, 118, 42, 29, 22, 15, 6, -2, -8, -12, -11, -8, -2, 4, 9, 11,
                11, 7, 2, -4, -9, -12, -11, -8, -2, 4, 10, 13, 12, 9, 3, -4, -10,
                -13, -13, -10, -4, 4, 10, 14, 15, 11, 5, -3, -10, -15, -16, -13,
                -6, 2, 10, 16, 17, 14, 8, -1, -10, -16, -19, -16, -9, 0, 10, 17,
                20, 18, 11, 1, -9, -18, -22, -20, -13, -3, 8, 18, 23, 22, 15, 5,
                -8, -18, -24, -24, -18, -7, 6, 18, 25, 26, 20, 9, -5, -18, -27,
                -28, -23, -11, 3, 18, 28, 30, 26, 14, -2, -17, -28, -33, -28, -17,
                -1, 16, 29, 35, 31, 20, 3, -15, -30, -37, -35, -23, -6, 14, 30,
                39, 38, 27, 9, -12, -30, -41, -41, -31, -12, 10, 30, 43, 45, 35,
                16, -8, -30, -45, -48, -39, -20, 5, 29, 46, 52, 44, 24, -2, -28,
                -48, -55, -49, -29, -2, 27, 49, 59, 54, 34, 6, -25, -50,
                -62, -59, -40, -11, 23, 51, 66, 65, 46, 16, -20, -51, -70, -71,
                -53, -22, 16, 51, 73, 77, 61, 28, -12, -51, -77, -83, -69, -36,
                8, 50, 80, 90, 77, 44, -2, -49, -84, -98, -87, -53, -4, 47, 87,
                106, 97, 63, 12, -44, -90, -114, -109, -75, -21, 40, 93, 123, 122,
                89, 31, -36, -96, -133, -136, -104, -44, 30, 99, 143, 152, 122, 58,
                -23, -101, -155, -171, -143, -76, 14, 103, 169, 194, 168, 97, -2,
                -105, -186, -221, -199, -124, -13, 107, 205, 255, 239, 159, 33,
                -109, -231, -299, -292, -207, -61, 110, 265, 360, 366, 275, 103,
                -112, -315, -452, -481, -383, -170, 112, 398, 611, 684, 580, 298,
                -113, -568, -953, -1152, -1067, -644, 114, 1140, 2312, 3475, 4458,
                5116, 5348, 5116, 4458, 3475, 2312, 1140, 114, -644, -1067, -1152,
                -953, -568, -113, 298, 580, 684, 611, 398, 112, -170, -383,
                -481, -452, -315, -112, 103, 275, 366, 360, 265, 110, -61, -207,
                -292, -299, -231, -109, 33, 159, 239, 255, 205, 107, -13, -124,
                -199, -221, -186, -105, -2, 97, 168, 194, 169, 103, 14, -76, -143,
                -171, -155, -101, -23, 58, 122, 152, 143, 99, 30, -44, -104, -136,
                -133, -96, -36, 31, 89, 122, 123, 93, 40, -21, -75, -109, -114,
                -90, -44, 12, 63, 97, 106, 87, 47, -4, -53, -87, -98, -84, -49, -2,
                44, 77, 90, 80, 50, 8, -36, -69, -83, -77, -51, -12, 28, 61, 77, 73,
                51, 16, -22, -53, -71, -70, -51, -20, 16, 46, 65, 66, 51, 23, -11,
                -40, -59, -62, -50, -25, 6, 34, 54, 59, 49, 27, -2, -29, -49, -55,
                -48, -28, -2, 24, 44, 52, 46, 29, 5, -20, -39, -48, -45, -30, -8, 16,
                35, 45, 43, 30, 10, -12, -31, -41, -41, -30,
                -12, 9, 27, 38, 39, 30, 14, -6, -23, -35, -37, -30, -15, 3, 20, 31,
                35, 29, 16, -1, -17, -28, -33, -28, -17, -2, 14, 26, 30, 28, 18, 3,
                -11, -23, -28, -27, -18, -5, 9, 20, 26, 25, 18, 6, -7, -18, -24,
                -24, -18, -8, 5, 15, 22, 23, 18, 8, -3, -13, -20, -22, -18, -9, 1,
                11, 18, 20, 17, 10, 0, -9, -16, -19, -16, -10, -1, 8, 14, 17, 16,
                10, 2, -6, -13, -16, -15, -10, -3, 5, 11, 15, 14, 10, 4, -4, -10,
                -13, -13, -10, -4, 3, 9, 12, 13, 10, 4, -2, -8, -11, -12, -9, -4,
                2, 7, 11, 11, 9, 4, -2, -8, -11, -12, -8, -2, 6, 15, 22, 29, 42, 118, -20};

volatile ioport unsigned int* SPCR1;
volatile ioport unsigned int* SPCR12;
volatile ioport unsigned int* SPCR2;
volatile ioport unsigned int* SRGR1;
volatile ioport unsigned int* SRGR2;
volatile ioport unsigned int* SRGR11;
volatile ioport unsigned int* SRGR21;
volatile ioport unsigned int* SRGR12;
volatile ioport unsigned int* SRGR22;
volatile ioport unsigned int* PCR;
volatile ioport unsigned int* PCR2;
volatile ioport unsigned int* TSSR;
volatile ioport unsigned int* PRD1;
volatile ioport unsigned int* PRD2;

/****************************************
 * Set the Codec configuration registers
 ****************************************/
void my_AIC23_rset(int hCodec, Uint16 regnum, Uint16 regval)
{
   /* Mask off lower 9 bits */
   regval &= 0x1ff;

   /* Wait for XRDY signal before writing data to DXR */
   while (!MCBSP_xrdy(C55XX_CONTROLHANDLE_hMcbsp));

   /* Write 16 bit data value to DXR */
   MCBSP_write16(C55XX_CONTROLHANDLE_hMcbsp, (regnum << 9) | regval);

   /* Save register value if regnum is in range */
   if (regnum < my_AIC23_NUMREGS)
      codecstate.regs[regnum] = regval;
}

/****************************************
 *  Configurate the Codec
 ****************************************/
void my_AIC23_config(int hCodec, my_AIC23_Config *Config)
{
   int i;

   /* Use default parameters if none are given */
   if (Config == NULL)
              Config = &config;

   /* Assign each register */
   for (i = 0; i < my_AIC23_NUMREGS; i++)
      my_AIC23_rset(hCodec, i, Config -> regs[i]);
}
/****************************************
 *  Write a 32-bit value to the codec
 ****************************************/
CSLBool my_AIC23_write32(Int32 val)
{
   /* Wait for XRDY signal before writing data to DXR */
   if(!MCBSP_xrdy(C55XX_DMA_MCBSP_hMcbsp)) {
      return (FALSE);
   }

   /* Write 32 bit data value to DXR, shift to match format mode */
   MCBSP_write32(C55XX_DMA_MCBSP_hMcbsp, val);

   return (TRUE);
}
/****************************************
 *  Write a 16-bit value to the codec
 ****************************************/
CSLBool my_AIC23_write16(Int16 val)
{
   /* Wait for XRDY signal before writing data to DXR */
   if(!MCBSP_xrdy(C55XX_DMA_MCBSP_hMcbsp)) {
      return (FALSE);
   }

   /* Write 32 bit data value to DXR, shift to match format mode */
   MCBSP_write16(C55XX_DMA_MCBSP_hMcbsp, val);

   return (TRUE);
}
/****************************************
 *  Write a 16-bit valueto the codec
 ****************************************/
CSLBool my_AIC23_read16(Int16 *val)
{
   /* Wait until data is ready then read */
   if (!MCBSP_rrdy(C55XX_DMA_MCBSP_hMcbsp)) {
      return (FALSE);
   }
   /* Read the data */
   *val = MCBSP_read16(C55XX_DMA_MCBSP_hMcbsp);
   return (TRUE);
}
/****************************************
 *  Read a 32-bit valueto the codec
 ****************************************/
CSLBool my_AIC23_read32(Int32 *val)
{
   /* Wait until data is ready then read */
   if (!MCBSP_rrdy(C55XX_DMA_MCBSP_hMcbsp)) {
      return (FALSE);
   }

   /* Read the data */
   *val = (Int32)MCBSP_read32(C55XX_DMA_MCBSP_hMcbsp);
   return (TRUE);
}
/*******************************************************************************
  ----------------------encode audio sample into IR word  -------------------
 *******************************************************************************/
void encodeIRword(Int16 AudioSample, Uint32 *IRtransLSB, Uint32 *IRtransMSB)
{
Uint32 IRwordLSB = 0x00000000, IRwordMSB = 0x00000000;

        if ((AudioSample & 0x0001) != 0) {IRwordLSB |= 0x00000002;}
        if ((AudioSample & 0x0002) != 0) {IRwordLSB |= 0x00000020;}
        if ((AudioSample & 0x0004) != 0) {IRwordLSB |= 0x00000200;}
        if ((AudioSample & 0x0008) != 0) {IRwordLSB |= 0x00002000;}
        if ((AudioSample & 0x0010) != 0) {IRwordLSB |= 0x00020000;}
        if ((AudioSample & 0x0020) != 0) {IRwordLSB |= 0x00200000;}
        if ((AudioSample & 0x0040) != 0) {IRwordLSB |= 0x02000000;}
        if ((AudioSample & 0x0080) != 0) {IRwordLSB |= 0x20000000;}

        if ((AudioSample & 0x0100) != 0) {IRwordMSB |= 0x00000002;}
        if ((AudioSample & 0x0200) != 0) {IRwordMSB |= 0x00000020;}
        if ((AudioSample & 0x0400) != 0) {IRwordMSB |= 0x00000200;}
        if ((AudioSample & 0x0800) != 0) {IRwordMSB |= 0x00002000;}
        if ((AudioSample & 0x1000) != 0) {IRwordMSB |= 0x00020000;}
        if ((AudioSample & 0x2000) != 0) {IRwordMSB |= 0x00200000;}
        if ((AudioSample & 0x4000) != 0) {IRwordMSB |= 0x02000000;}
        if ((AudioSample & 0x8000) != 0) {IRwordMSB |= 0x20000000;}
        *IRtransLSB = IRwordLSB;
        *IRtransMSB = IRwordMSB;
}

/*******************************************************************************
  ----------------------decode IR word into audio sample  -------------------
 *******************************************************************************/
void decodeIRword(Int16 *AudioSample, Uint32 recieved_wordLSB, Uint32 recieved_wordMSB)
{
Int16   HdPhoneSample = 0x0000;

        if ((recieved_wordLSB & 0x00000007) != 0) {HdPhoneSample |= 0x0001;}
        if ((recieved_wordLSB & 0x00000070) != 0) {HdPhoneSample |= 0x0002;}
        if ((recieved_wordLSB & 0x00000700) != 0) {HdPhoneSample |= 0x0004;}
        if ((recieved_wordLSB & 0x00007000) != 0) {HdPhoneSample |= 0x0008;}
        if ((recieved_wordLSB & 0x00070000) != 0) {HdPhoneSample |= 0x0010;}
        if ((recieved_wordLSB & 0x00700000) != 0) {HdPhoneSample |= 0x0020;}
        if ((recieved_wordLSB & 0x07000000) != 0) {HdPhoneSample |= 0x0040;}
        if ((recieved_wordLSB & 0x70000000) != 0) {HdPhoneSample |= 0x0080;}

        if ((recieved_wordMSB & 0x00000007) != 0) {HdPhoneSample |= 0x0100;}
        if ((recieved_wordMSB & 0x00000070) != 0) {HdPhoneSample |= 0x0200;}
        if ((recieved_wordMSB & 0x00000700) != 0) {HdPhoneSample |= 0x0400;}
        if ((recieved_wordMSB & 0x00007000) != 0) {HdPhoneSample |= 0x0800;}
        if ((recieved_wordMSB & 0x00070000) != 0) {HdPhoneSample |= 0x1000;}
        if ((recieved_wordMSB & 0x00700000) != 0) {HdPhoneSample |= 0x2000;}
        if ((recieved_wordMSB & 0x07000000) != 0) {HdPhoneSample |= 0x4000;}
        if ((recieved_wordMSB & 0x70000000) != 0) {HdPhoneSample |= 0x8000;}
        *AudioSample = HdPhoneSample;
}


/*******************************************************************************
  ----------------------Comms Restart Frame  -------------------
 ******************************************************************************/
void TINT0_HWI(void)
{
            asm(" NOP");
            asm(" NOP");
            *SPCR1 = 0x0000;                //put MCBSP reciever into reset
            asm(" NOP");
            asm(" NOP");
            *PCR = 0x0A00;
            asm(" NOP");
            asm(" NOP");

            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00000000);

            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00008001);   //must be 8001 or get preamble problems
            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp, Headset_code);//send base headset pair code

            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00000000);
            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00000000);
            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00000000);
            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00000000);

   GPT_stop12(hGpt);                //need to reinitialize PRD registers, since RINT0 ISR could make
            *PRD1 = 0x00EC;     //make period too long for comms restart
            *PRD2 = 0x0002;
            GPT_start12(hGpt);

            asm(" NOP");
            asm(" NOP");
            *PCR = 0x0B00;
            asm(" NOP");
            asm(" NOP");
            *SPCR1 = 0x0001;                //take MCBSP reciever out of reset
            asm(" NOP");
            asm(" NOP");
}

/******************************************************************************
  ---------Recieve Headset microphone sample and send to codec  -------------
 ******************************************************************************/
void IRdata_recievedHWI(void)
{
            while (!MCBSP_xrdy(mhMcbsp));           //make sure Tx is off
            MCBSP_write16(mhMcbsp,0x0000);

            while (!MCBSP_rrdy(mhMcbsp));
            dummy = MCBSP_read32(mhMcbsp);

            while (!MCBSP_rrdy(mhMcbsp));
            RcvCode = MCBSP_read32(mhMcbsp);
            while (!MCBSP_rrdy(mhMcbsp));
            recieved_wordMsb[0] = MCBSP_read32(mhMcbsp);

            while (!MCBSP_rrdy(mhMcbsp));
            dummy = MCBSP_read32(mhMcbsp);

            while (!MCBSP_rrdy(mhMcbsp));
            recieved_wordMsb[1] = MCBSP_read32(mhMcbsp);
            while (!MCBSP_rrdy(mhMcbsp));
            recieved_wordMsb[2] = MCBSP_read32(mhMcbsp);

            while (!MCBSP_rrdy(mhMcbsp));
            dummy = MCBSP_read32(mhMcbsp);

            while (!MCBSP_rrdy(mhMcbsp));
            recieved_wordLsb[0] = MCBSP_read32(mhMcbsp);
            while (!MCBSP_rrdy(mhMcbsp));
            recieved_wordLsb[1] = MCBSP_read32(mhMcbsp);
```

```
        while (!MCBSP_rrdy(mhMcbsp));
        dummy = MCBSP_read32(mhMcbsp);

        while (!MCBSP_rrdy(mhMcbsp));
        recieved_wordLsb[2] = MCBSP_read32(mhMcbsp);

        while (!MCBSP_rrdy(mhMcbsp));
        dummy = MCBSP_read32(mhMcbsp);
        while (!MCBSP_rrdy(mhMcbsp));
        dummy = MCBSP_read32(mhMcbsp);

        IRQ_globalDisable();
        while (!MCBSP_xrdy(mhMcbsp));           //make sure Tx is off
        MCBSP_write16(mhMcbsp,0x0000);

        asm(" NOP");
        asm(" NOP");
        *SPCR1 = 0x0000;                //put MCBSP reciever into reset
        asm(" NOP");
        asm(" NOP");
        *PCR = 0x0A00;
        asm(" NOP");
        asm(" NOP");

        //-----------------------get sample's synced to RINT0 ISR-------------

dmastat2 = DMA_RGETH(hDmaRcv2, DMACDAC);              //Read DMA status register to clear any errors
        dmastat1 = DMA_RGETH(hDmaRcv1, DMACDAC);

        ElmtL = dmastat2/2 - AudioInSampleLaddr;
        ElmtR = dmastat1/2 - AudioInSampleRaddr;
ElmtLp2 = ElmtL +2;
        ElmtRp2 = ElmtR +2;
ElmtLm18 = ElmtL -18;
        ElmtRm18 = ElmtR -18;

for (j=0; j<18; j++) {//allow DMA to restart sooner than if waiting for FIR
                if ((ElmtLm18 +j) < 0) { AudioInSampleLi[j] = AudioInSampleL[ElmtLp2 +j];}
                else {AudioInSampleLi[j] = AudioInSampleL[ElmtLm18 +j];}

                if ((ElmtRm18 +j) < 0) { AudioInSampleRi[j] = AudioInSampleR[ElmtRp2 +j]; }
                else {AudioInSampleRi[j] = AudioInSampleR[ElmtRm18 +j];}
                }

        if (AudioInSampleLi[0] == AudioInSampleLold)  //if headset faster than base
                        {AudioInSampleLi[0] = AudioInSampleLi[1];}//make it repeat last samples
        if (AudioInSampleRi[0] == AudioInSampleRold)
                        {AudioInSampleRi[0] = AudioInSampleRi[1];}
        if (AudioInSampleLi[17] == AudioInSampleLold)  //if headset faster than base
                        {AudioInSampleLi[17] = AudioInSampleLi[16];}//make it repeat last samples
        if (AudioInSampleRi[17] == AudioInSampleRold)
                        {AudioInSampleRi[17] = AudioInSampleRi[16];}


        //---------------------------------------------------------------------

        BaseOutSampleOld[0] = BaseOutSamplei[0];              //store previous audio sample values
        BaseOutSampleOld[1] = BaseOutSamplei[1];
        BaseOutSampleOld[2] = BaseOutSamplei[2];

        decodeIRword(&BaseOutSamplei[0], recieved_wordLsb[0], recieved_wordMsb[0]);
        decodeIRword(&BaseOutSamplei[1], recieved_wordLsb[1], recieved_wordMsb[1]);
        decodeIRword(&BaseOutSamplei[2], recieved_wordLsb[2], recieved_wordMsb[2]);

        dummy = 0;
        decodeIRword(&CdChk, RcvCode, dummy);

        crc8_codeRcv = 0;
        for (i=0 ; i<3 ; i=i+1)
          {
          tempRcv = ((BaseOutSamplei[i]) & 0xF000) >> 8;
          indexRcv = ((Uchar) tempRcv) ^ crc8_codeRcv;
          crc8_codeRcv = crc8_data[indexRcv];
          }
        crc8_codeRcv = crc8_codeRcv &0xF0;
        CdChk = CdChk &0xF0;
        token = 0;
        if ((abs(BaseOutSampleOld[0] - BaseOutSamplei[0])) > limit) {token = 1;}
        if ((abs(BaseOutSampleOld[1] - BaseOutSamplei[1])) > limit) {token = 1;}
        if ((abs(BaseOutSampleOld[2] - BaseOutSamplei[2])) > limit) {token = 1;}

        //if limit is too low then can get glithes even when there are no errors
        if (abs(BaseOutSamplei[0]) < 20000) { token = 0;}

        //              Make Headset not transmit if it detects errors, fixes problem
        if ((crc8_codeRcv != CdChk) || token == 1) {  //compare the generated and transmitted CRC checksums
                BaseOutSamplei[0] = BaseOutSampleOld[2];                        //and flatline if errors are detected
                BaseOutSamplei[1] = BaseOutSampleOld[2];
                BaseOutSamplei[2] = BaseOutSampleOld[2];
                limit = 5000;                   //if an error occurred in the last transmission limit volume variation to
half peak
                }
        else { limit = 32700;} //if no error occured in last transmission allow full volume range

        DMA_RGETH(hDmaXmt, DMACSR);
        BaseOutSamplei[5] = BaseOutSamplei[2];
        BaseOutSamplei[4] = BaseOutSamplei[2];
        BaseOutSamplei[3] = BaseOutSamplei[1];
        BaseOutSamplei[2] = BaseOutSamplei[1];
        BaseOutSamplei[1] = BaseOutSamplei[0];
        BaseOutSamplei[0] = BaseOutSamplei[0];



  BaseOutSample[0]  = BaseOutSamplei[0];                //store previous audio sample values
        BaseOutSample[1]  = BaseOutSamplei[1];
        BaseOutSample[2]  = BaseOutSamplei[2];
        BaseOutSample[3]  = BaseOutSamplei[3];
        BaseOutSample[4]  = BaseOutSamplei[4];
        BaseOutSample[5]  = BaseOutSamplei[5];
        BaseOutSample[6]  = BaseOutSamplei[5];
        BaseOutSample[7]  = BaseOutSamplei[5];

        //---------------------------- Send samples to headset ---------------------------
        // Put Audio through low pass filter. First left channel, then right channel, but it dulls the speech
  firdec(AudioInSampleLi, COEFFS, LPAudioInSampleL, delayBufferL, ORDER, 18, 6);
  firdec(AudioInSampleRi, COEFFS, LPAudioInSampleR, delayBufferR, ORDER, 18, 6);

        while (!MCBSP_xrdy(mhMcbsp));           //make sure Tx is off
        MCBSP_write16(mhMcbsp,0x0000);

        ErrCode = 0x00000000;
        encodeIRword(LPAudioInSampleL[0], &transmit_wordLlsb[0], &transmit_wordLmsb[0]);
        encodeIRword(LPAudioInSampleR[0], &transmit_wordRlsb[0], &transmit_wordRmsb[0]);
        encodeIRword(LPAudioInSampleL[1], &transmit_wordLlsb[1], &transmit_wordLmsb[1]);
        encodeIRword(LPAudioInSampleR[1], &transmit_wordRlsb[1], &transmit_wordRmsb[1]);
        encodeIRword(LPAudioInSampleL[2], &transmit_wordLlsb[2], &transmit_wordLmsb[2]);
        encodeIRword(LPAudioInSampleR[2], &transmit_wordRlsb[2], &transmit_wordRmsb[2]);

  crc8_codeTx = 0;
  for (i=0 ; i<3 ; i++)
          {
          tempTx = ((LPAudioInSampleL[i]) & 0xF000) >> 8;
          indexTx = ((Uchar) tempTx) ^ crc8_codeTx;
          crc8_codeTx = crc8_data[indexTx];

          tempTx = ((LPAudioInSampleR[i]) & 0xF000) >> 8;
          indexTx = ((Uchar) tempTx) ^ crc8_codeTx;
          crc8_codeTx = crc8_data[indexTx];
          }
        encodeIRword(crc8_codeTx, &ErrCode, &dummy);

        GPT_getCnt(hGpt,tim34,tim12);
  GPT_stop12(hGpt);
        if (*tim12 > 0xFFFDB000) { PeriodValue = 0x200EC;}
        else {PeriodValue = *tim12 + 0x200EC;}
        t1 = PeriodValue & 0x0000FFFF;
        t2 = PeriodValue / 0x10000;
        *PRD1 = t1;
        *PRD2 = t2;
        GPT_start12(hGpt);
```

```
for (i=0 ; i<10 ; i++) {
                while (!MCBSP_xrdy(mhMcbsp));               //delay for at least 50us
                MCBSP_write32(mhMcbsp,0x00000000);
                }

transmit_wordLmsb[0] &= 0x22222222;      //make sure 1/4 modulation is observed
transmit_wordLlsb[0] &= 0x22222222;       //it is somehow violated with code so need this
transmit_wordLmsb[1] &= 0x22222222;
transmit_wordLlsb[1] &= 0x22222222;
transmit_wordLmsb[2] &= 0x22222222;
transmit_wordLlsb[2] &= 0x22222222;
transmit_wordRmsb[0] &= 0x22222222;
transmit_wordRlsb[0] &= 0x22222222;
transmit_wordRmsb[1] &= 0x22222222;
transmit_wordRlsb[1] &= 0x22222222;
transmit_wordRmsb[2] &= 0x22222222;
transmit_wordRlsb[2] &= 0x22222222;
ErrCode &= 0x22222222;

while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,0x00000000);                  //make sure transciever LED turns off

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,0x00000080);                  //make sure transciever LED turns off

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,0xC0000000);                  //make sure transciever LED turns off

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp, Headset_code);//send base headset pair code
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordLmsb[0]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordLlsb[0]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordRmsb[0]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordRlsb[0]);

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordLmsb[1]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordLlsb[1]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordRmsb[1]);
// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordRlsb[1]);

// Wait for XRDY signal before writing data to DXR
while (!MCBSP_xrdy(mhMcbsp));
// Write 32 bit data value to DXR
MCBSP_write32(mhMcbsp,transmit_wordLmsb[2]);
```

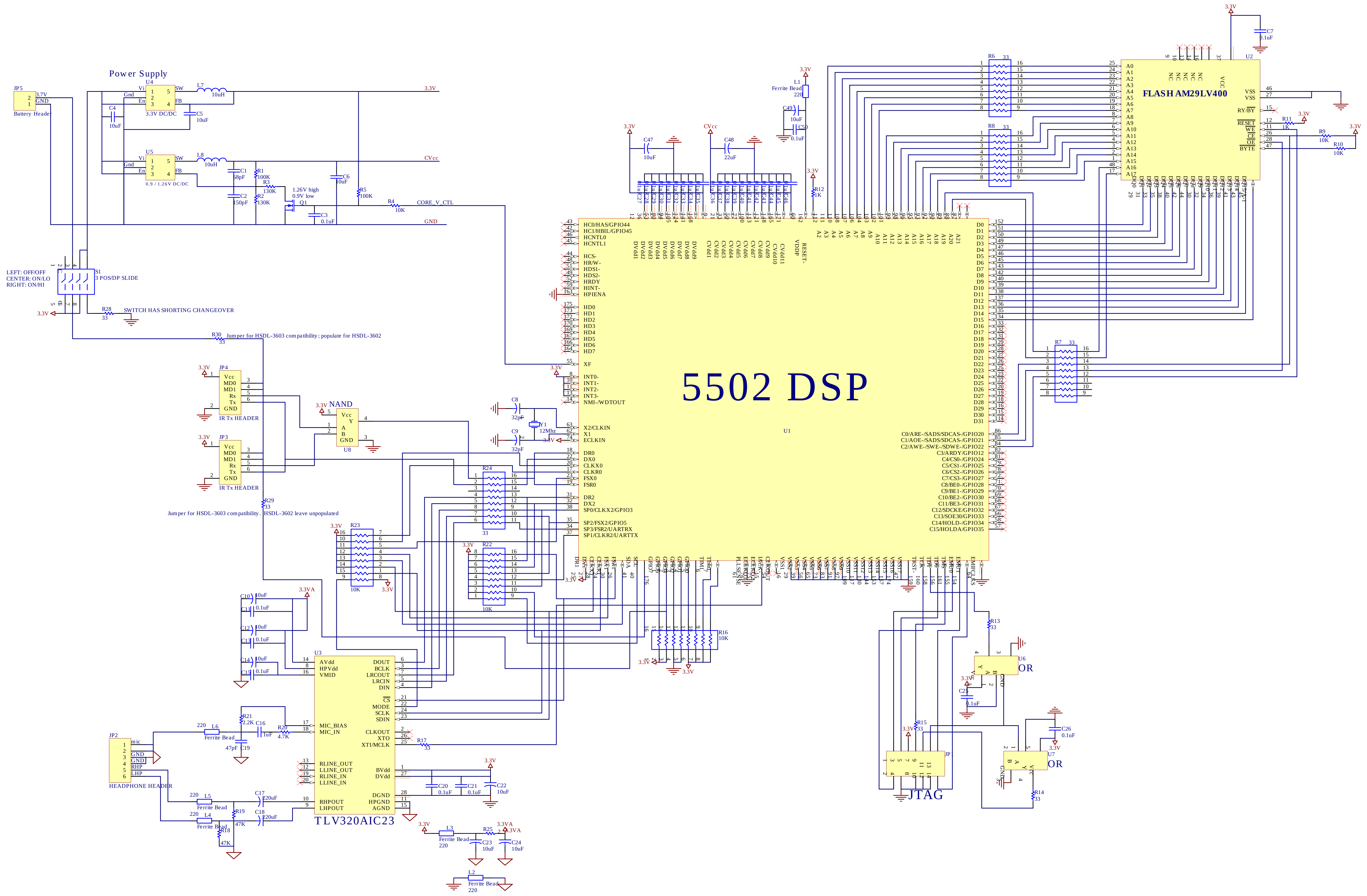
```
		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,transmit_wordLlsb[2]);
		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,transmit_wordRmsb[2]);
		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,transmit_wordRlsb[2]);

		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,ErrCode);

		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,0x00000000);		//put a clear word spacing between words
		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,0x00000000);		//put a clear word spacing between words
		// Wait for XRDY signal before writing data to DXR
		while (!MCBSP_xrdy(mhMcbsp));
		// Write 32 bit data value to DXR
		MCBSP_write32(mhMcbsp,0x00000000);		//put a clear word spacing between words

		asm(" NOP");
		asm(" NOP");
		*PCR = 0x0B00;
		asm(" NOP");
		asm(" NOP");
		*SPCR1 = 0x0001;		//take MCBSP reciever out of reset
		asm(" NOP");
		asm(" NOP");

		AudioInSampleLold = AudioInSampleLi[17];
		AudioInSampleRold = AudioInSampleRi[17];

		while (!MCBSP_xrdy(mhMcbsp));
		MCBSP_write16(mhMcbsp,0x0000);
		IRQ_globalEnable();
		HWI_enable();
}

/*******************************************************************************
 *			======== MAIN ========
 *******************************************************************************/
Void main()
{
		SPCR1 = (volatile ioport unsigned int*)0x2804;
		SPCR12 = (volatile ioport unsigned int*)0x3004;
		SPCR2 = (volatile ioport unsigned int*)0x2805;
		SRGR1 = (volatile ioport unsigned int*)0x280A;
		SRGR2 = (volatile ioport unsigned int*)0x280B;
		SRGR11 = (volatile ioport unsigned int*)0x2C0A;
		SRGR21 = (volatile ioport unsigned int*)0x2C0B;
		SRGR12 = (volatile ioport unsigned int*)0x300A;
		SRGR22 = (volatile ioport unsigned int*)0x300B;
		PCR = (volatile ioport unsigned int*)0x2812;
		PCR2 = (volatile ioport unsigned int*)0x3012;
   TSSR = (volatile ioport unsigned int*)0x8000;
   PRD1 = (volatile ioport unsigned int*)0x100C;
   PRD2 = (volatile ioport unsigned int*)0x100D;

		//Initialize CSL library
		CSL_init();

   PLL_config(&MyPLLConfig);
		EMIF_RSET(GBLCTL2, 0x5);  //make codec clock 3MHz

		//Set the required frequncy for CPU, Fast and Slow peripherals and EMIF
```

```
            PLL_setFreq(1,8,0,0,0,3,0);

            // Open MCBSP Port 0, thiswill return a MCBSP handle that will be used in calls to other CSl functions.
            mhMcbsp = MCBSP_open(MCBSP_PORT0, MCBSP_OPEN_RESET);

            *SRGR1 = 0x010B; //SRGR1, SRGR2 do not get set by config, so do manually, are before cofig
            *SRGR2 = 0x200F; // because to change McBSP settings, port must be reset
            // Write configuration structure values to MCBSP control registers
            MCBSP_config(mhMcbsp, &ConfigLoopBack320);

             /* Start Sample Rate Generator and Frame Sync */
            MCBSP_start(mhMcbsp, MCBSP_SRGR_START | MCBSP_SRGR_FRAMESYNC,0x300);

            /* Enable MCBSP transmit and receive */
            MCBSP_start(mhMcbsp, MCBSP_RCV_START | MCBSP_XMIT_START, 0x200);

   // Open MCBSP Port 1, this will return a MCBSP handle that will be used in calls to other CSl functions.
            C55XX_CONTROLHANDLE_hMcbsp = MCBSP_open(MCBSP_PORT1, MCBSP_OPEN_RESET);
            *SRGR11 = 0x0063; //SRGR1, SRGR2 do not get set by config, so do manually, are before cofig
            *SRGR21 = 0x2013;
            // Write configuration structure values to MCBSP control registers
            MCBSP_config(C55XX_CONTROLHANDLE_hMcbsp, &ConfigMcbsp1);

            // Open MCBSP Port 2, this will return a MCBSP handle that will be used in calls to other CSl functions.
            C55XX_DMA_MCBSP_hMcbsp = MCBSP_open(MCBSP_PORT2, MCBSP_OPEN_RESET);
            *SRGR12 = 0x0000; //SRGR1, SRGR2 do not get set by config, so do manually, are before cofig
            *SRGR22 = 0x2000;
            // Write configuration structure values to MCBSP control registers
            MCBSP_config(C55XX_DMA_MCBSP_hMcbsp, &ConfigMcbsp2);

   // Start McBSP1 as the codec control channel
   MCBSP_start(C55XX_CONTROLHANDLE_hMcbsp,     MCBSP_XMIT_START     |     MCBSP_SRGR_START     |
MCBSP_SRGR_FRAMESYNC, 100);

   // Reset the AIC23
   my_AIC23_rset(0, my_AIC23_RESET, 0);

   // Configure the rest of the AIC23 registers */
   my_AIC23_config(0, &config);

            /* Clear any garbage from the codec data port */
   if (MCBSP_rrdy(C55XX_DMA_MCBSP_hMcbsp))
     MCBSP_read16(C55XX_DMA_MCBSP_hMcbsp);

   // Start McBSP2 as the codec data channel
   MCBSP_start(C55XX_DMA_MCBSP_hMcbsp, MCBSP_XMIT_START | MCBSP_RCV_START |
            MCBSP_SRGR_START | MCBSP_SRGR_FRAMESYNC, 0x300);

   /* Open DMA Channel 0 */
   hDmaXmt = DMA_open(DMA_CHA0, 0);
   hDmaRcv1 = DMA_open(DMA_CHA1, 0);
   hDmaRcv2 = DMA_open(DMA_CHA2, 0);

   /* By default, the TMS320C55xx compiler assigns all data symbols word */
   /* addresses. The DMA however, expects all addresses to be byte      */
   /* addresses. Therefore, we must shift the address by 2 in order to   */
   /* change the word address to a byte address for the DMA transfer.    */
   srcAddrHi = (Uint16)(((Uint32)(MCBSP_ADDR(DRR12))) >> 15) & 0xFFFFu;
   srcAddrLo = (Uint16)(((Uint32)(MCBSP_ADDR(DRR12))) << 1) & 0xFFFFu;
   dstAddrHi = (Uint16)(((Uint32)(&AudioInSampleR[0])) >> 15) & 0xFFFFu;
   dstAddrLo = (Uint16)(((Uint32)(&AudioInSampleR[0])) << 1) & 0xFFFFu;
   dmaRcvConfig1.dmacssal = (DMA_AdrPtr)srcAddrLo;
   dmaRcvConfig1.dmacssau = srcAddrHi;
   dmaRcvConfig1.dmacdsal = (DMA_AdrPtr)dstAddrLo;
   dmaRcvConfig1.dmacdsau = dstAddrHi;

   srcAddrHi = (Uint16)(((Uint32)(MCBSP_ADDR(DRR22))) >> 15) & 0xFFFFu;
   srcAddrLo = (Uint16)(((Uint32)(MCBSP_ADDR(DRR22))) << 1) & 0xFFFFu;
   dstAddrHi = (Uint16)(((Uint32)(&AudioInSampleL[0])) >> 15) & 0xFFFFu;
   dstAddrLo = (Uint16)(((Uint32)(&AudioInSampleL[0])) << 1) & 0xFFFFu;
   dmaRcvConfig2.dmacssal = (DMA_AdrPtr)srcAddrLo;
   dmaRcvConfig2.dmacssau = srcAddrHi;
   dmaRcvConfig2.dmacdsal = (DMA_AdrPtr)dstAddrLo;
   dmaRcvConfig2.dmacdsau = dstAddrHi;

            srcAddrHi = (Uint16)(((Uint32)(&BaseOutSample[0])) >> 15) & 0xFFFFu;
   srcAddrLo = (Uint16)(((Uint32)(&BaseOutSample[0])) << 1) & 0xFFFFu;
   dstAddrHi = (Uint16)(((Uint32)(MCBSP_ADDR(DXR12))) >> 15) & 0xFFFFu;
   dstAddrLo = (Uint16)(((Uint32)(MCBSP_ADDR(DXR12))) << 1) & 0xFFFFu;
   dmaXmtConfig.dmacssal = (DMA_AdrPtr)srcAddrLo;
   dmaXmtConfig.dmacssau = srcAddrHi;
   dmaXmtConfig.dmacdsal = (DMA_AdrPtr)dstAddrLo;
   dmaXmtConfig.dmacdsau = dstAddrHi;

   /* Write configuration structure values to DMA control registers */
   DMA_config(hDmaXmt, &dmaXmtConfig);
   DMA_config(hDmaRcv2, &dmaRcvConfig2);
   DMA_config(hDmaRcv1, &dmaRcvConfig1);

   /* Enable DMA channel to begin transfer */
   DMA_start(hDmaRcv2);
   DMA_start(hDmaRcv1);
   DMA_start(hDmaXmt);

            //get physical address of arrays for use in circular buffers, by trial and error reached this statement that works
            AudioInSampleLaddr = (Uint16)((Uint32)((DMA_AdrPtr)&AudioInSampleL[0]));
            AudioInSampleRaddr = (Uint16)((Uint32)((DMA_AdrPtr)&AudioInSampleR[0]));
            BaseOutSampleAddr = (Uint16)((Uint32)((DMA_AdrPtr)&BaseOutSample[0]));

            *TSSR = 0x0038;                    //Set Timer pins to output
            hGpt = GPT_open(GPT_DEV0, GPT_OPEN_RESET);
            GPT_config(hGpt, &MyConfig);
            GPT_start12(hGpt);

            while((!my_AIC23_write32(0x00000000)) && (delay < 10)) {delay++;}   //must clear any garbage from McBSP
port for DMA to work
            delay = 0;
            while((!my_AIC23_read16(&BaseOutSample[0])) && (delay < 1000)) {delay++;}  //must clear any garbage from
McBSP port for DMA to work
            delay = 0;

            if (delay == 5) {
                        IRdata_recievedHWI();
                        TINT0_HWI();
                        }

            while (!MCBSP_xrdy(mhMcbsp));
            MCBSP_write32(mhMcbsp,0x00000000);

   IRQ_enable(IRQ_EVT_RINT0);          //enable in intm register
   IRQ_enable(IRQ_EVT_TINT0);          //enable in intm register
            HWI_enable();

   IDL_run();
}
```

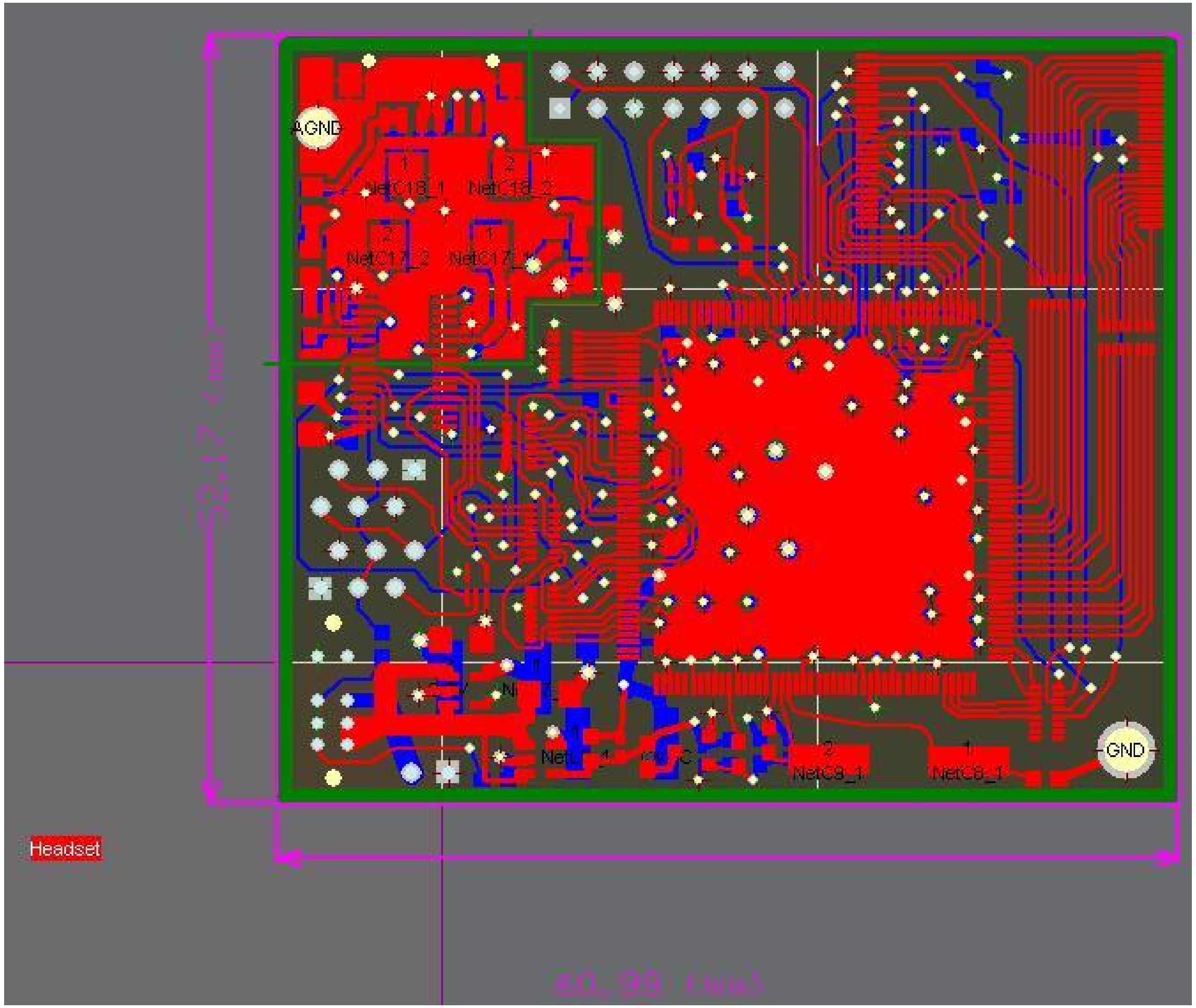
Appendix III. Prototype Circuit Design

Application Software - Protel DXP Electronic Design Package

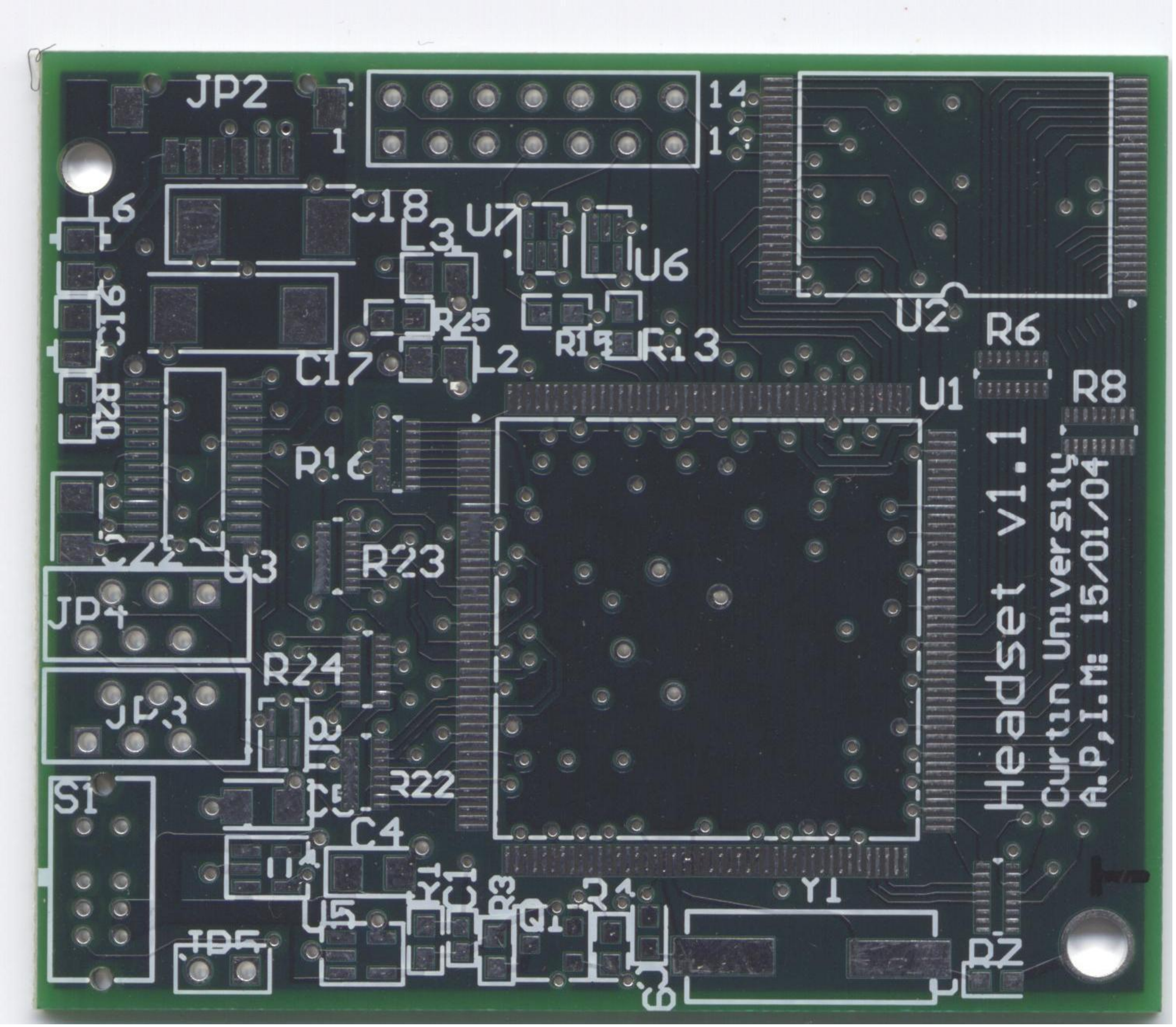
III.a. Headset Circuit Schematic



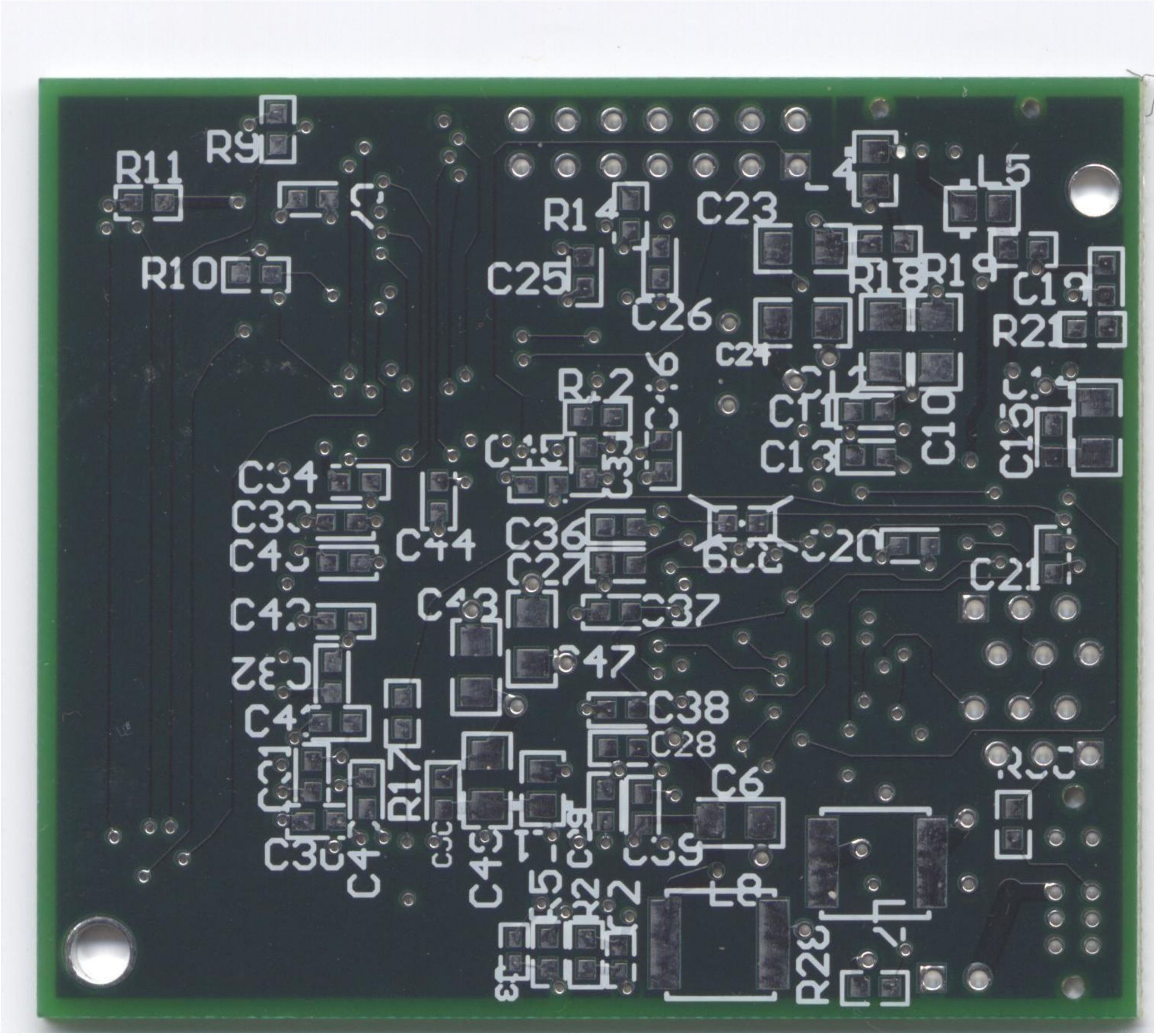
III.b. Headset PCB Layout



III.c. Headset Unpopulated PCB - Top View



III.d. Headset Unpopulated PCB - Bottom View

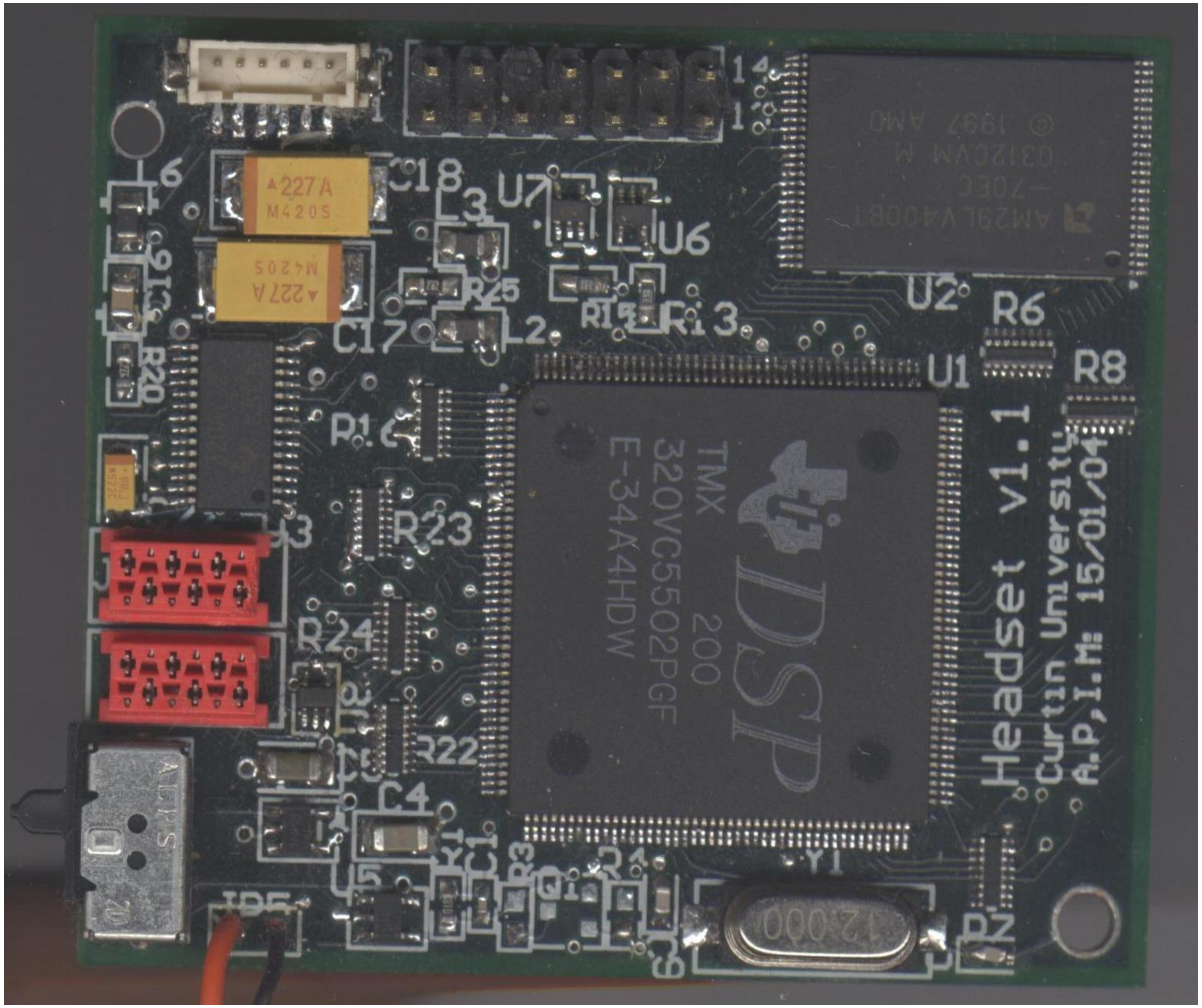


III.e. Headset Bill of Materials

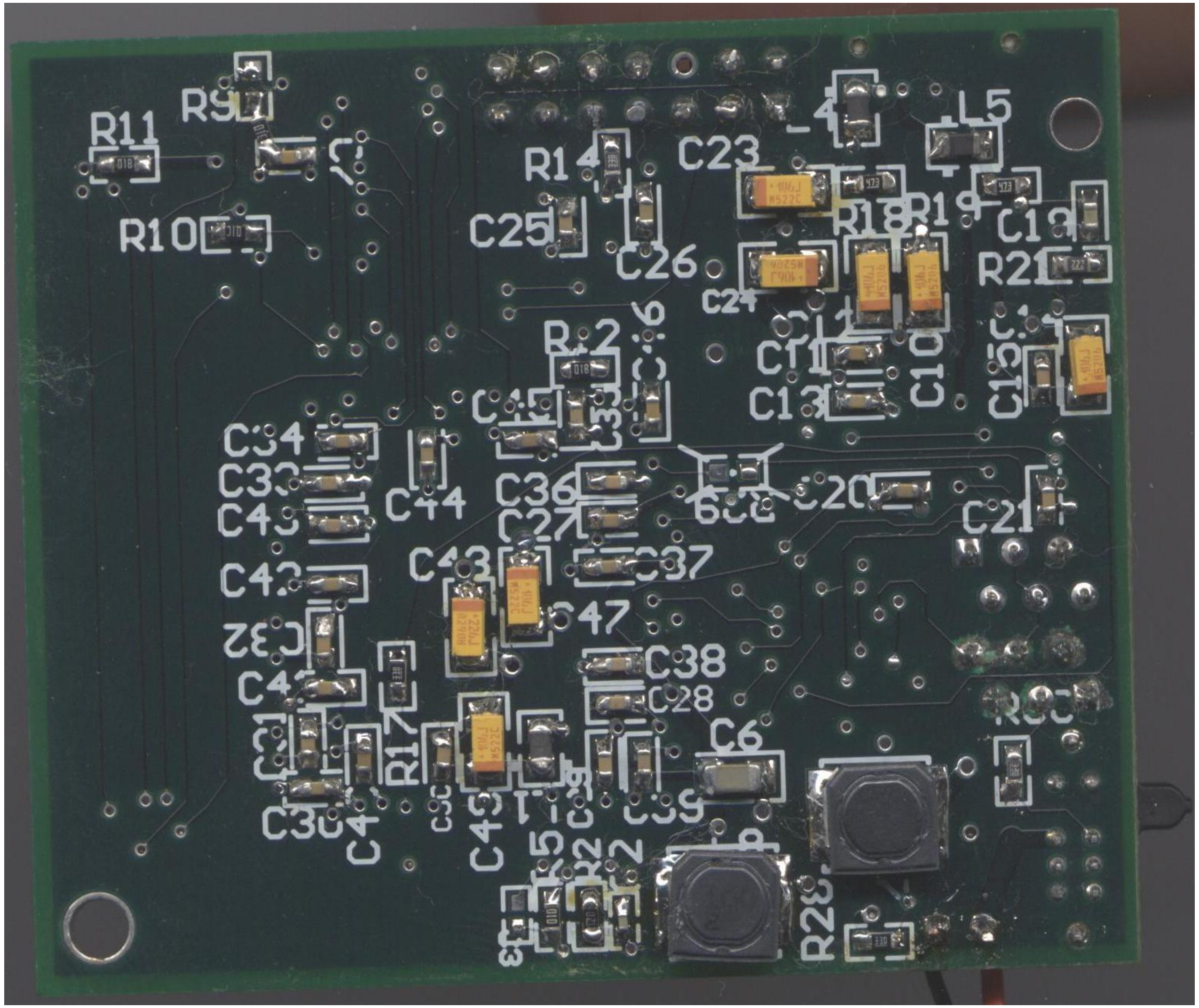
Description	Designator	Footprint	Quantity
CAP, CER, 0402, 68pF	C1	C1005-0402	1
CAP, CER, 0402, 150pF	C2	C1005-0402	1
CAP, CER, 0402, X7R, 0.1uF	C3	C1005-0402	1
CAP, CER, 0805 10uF	C4, C5, C6	CC3216-1206	3
CAP, CER, 0603, X7R, 0.1uF	C7, C11, C13, C15, C20, C21, C25, C26, C27, C28, C29, C30, C31, C32, C33,	CC1608-0603	29
	C34, C35, C36, C37, C38, C39, C40, C41, C42, C43, C44, C45, C46, C50		
CAP,TANT, 0805, 32pF	C8, C9	CC1608-0603	2
CAP,TANT, 1206, 10uF	C10, C12, C14, C22, C23, C24, C47, C48, C49	C3216-1206, CC3216-1206	9
CAP,CER,SMT 0603,1uF,6.3V,X5R,+/-10%	C16	CC2012-0805	1
CAP,TANT,SMT,220uF,10V	C17, C18	Cap 220uF	2
CAP,CER,SMT 0603,47pF,50V,+/-5%,NPO	C19	CC1608-0603	1
Header, 7-Pin, Dual row	JP1	JTAG	1
Header, 6-Pin	JP2, JP3, JP4	Headphone Header, MicroMatch 6-way	3
Header, 2-Pin	JP5	HDR1X2	1
FERRITE BEAD,SMT 0805,220 OHMS	L1, L2, L3, L4, L5, L6	CC2012-0805	6
INDUCTOR,SMT,10uH	L7, L8	Inductor panasonic ELL6X	2
N-Channel MOSFET	Q1	BSS138	1
RES,SMT 0805,200K OHM,1%,1/10 WATT	R1	CR1608-0603	1
RES,SMT 0805,130K OHM,1%,1/10 WATT	R2	CR1608-0603	1
Resistor	R3, R25	CR1608-0603	2
RES,SMT 0603,10K OHM,5%,1/16 WATT	R4, R10	CR1608-0603	2
RES,SMT 0603,100K OHM,1%,1/16 WATT	R5	CR1608-0603	1
Resistor Array-8, 33 OHM	R6, R8, R24	new r8 pack	3
Resistor Array-4, 33 OHM	R7	new r8 pack	1
RES,SMT 0603,1K OHM,5%,1/16 WATT	R9, R11, R12	CR1608-0603	3
RES,SMT 0603,33 OHM,5%,1/16 WATT	R13, R14, R15, R17, R28	CR1608-0603	5
Resistor Array-8, 10K OHM	R16, R22, R23	new r8 pack	3
RES,SMT 0603,47K OHM,5%,1/16 WATT	R18, R19	CR1608-0603	2
RES,SMT 0603,4.7K OHM,5%,1/16 WATT	R20	CR1608-0603	1
RES,SMT 0603,2.2K OHM,5%,1/16 WATT	R21	CR1608-0603	1
DIP Switch	S1	Slide Switch SSAC120200	1
5502 DSP	U1	F-QFP24x24-G176/N	1
IC, AMD Flash Memory, AM29LV400	U2	TSSO12x20-G48/P.5	1
IC, CODEC TLV320AIC23	U3	TSSO10x6-G28	1
IC, 3.3V DC/DC Regulator, 62203	U4	SO-G5/P.95	1
IC, DC/DC Regulator, 62200	U5	SO-G5/P.95	1
IC,SO5,SINGLE 2-INPUT POSITIVE-OR GATE SN74LVC1G32DCKR	U6, U7	SO-G5/P.65	2

Description	Designator	Footprint	Quantity
IC,SO5,SINGLE 2-INPUT NAND GATE SN74LVC1G32DCKRSN74LVC1G00DCKR	U8	SO-G5/P.65	1
Crystal Oscillator	Y1	NLS7 CRYSTAL	1

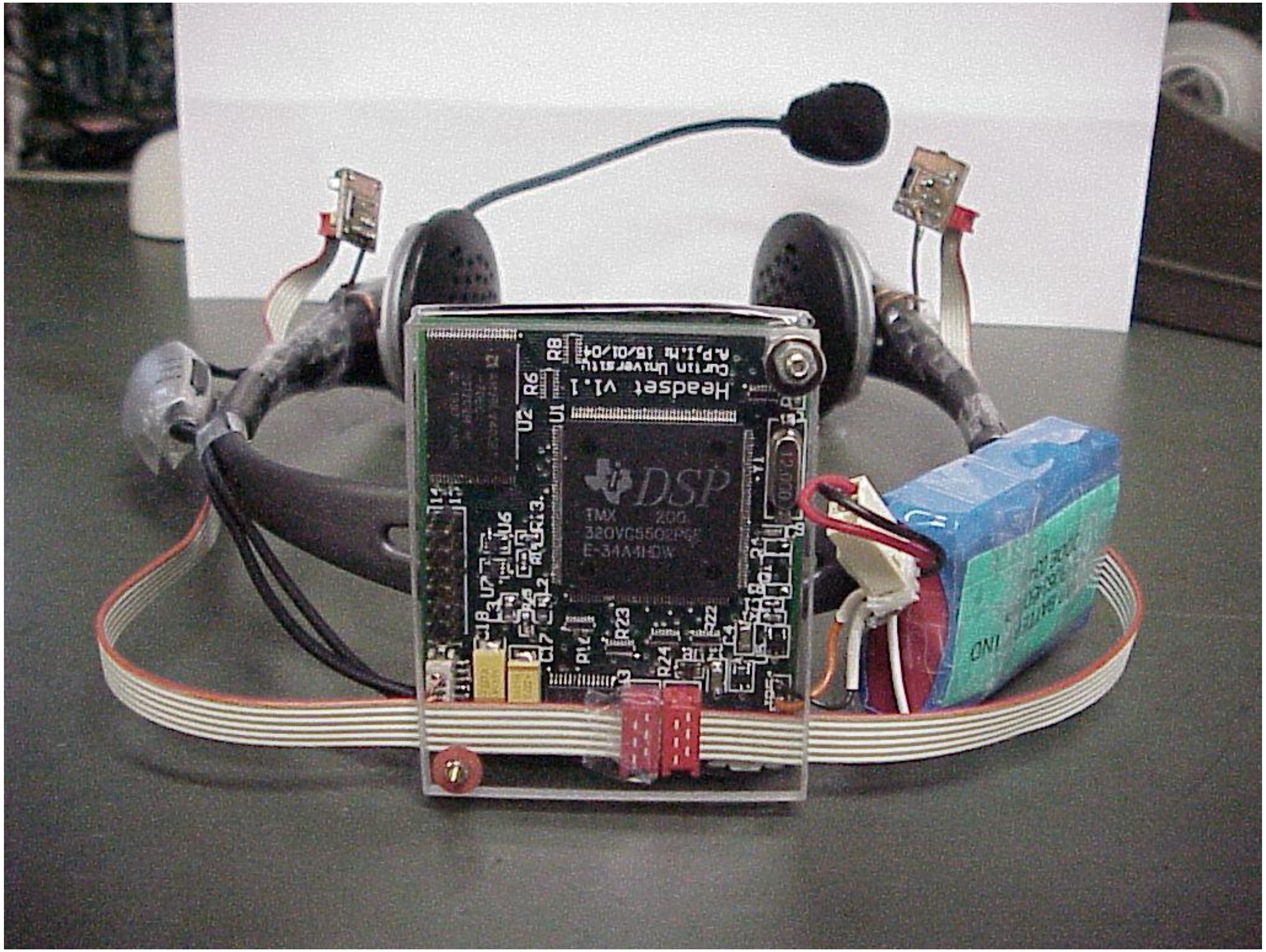
III.f. Headset PCB - Top View

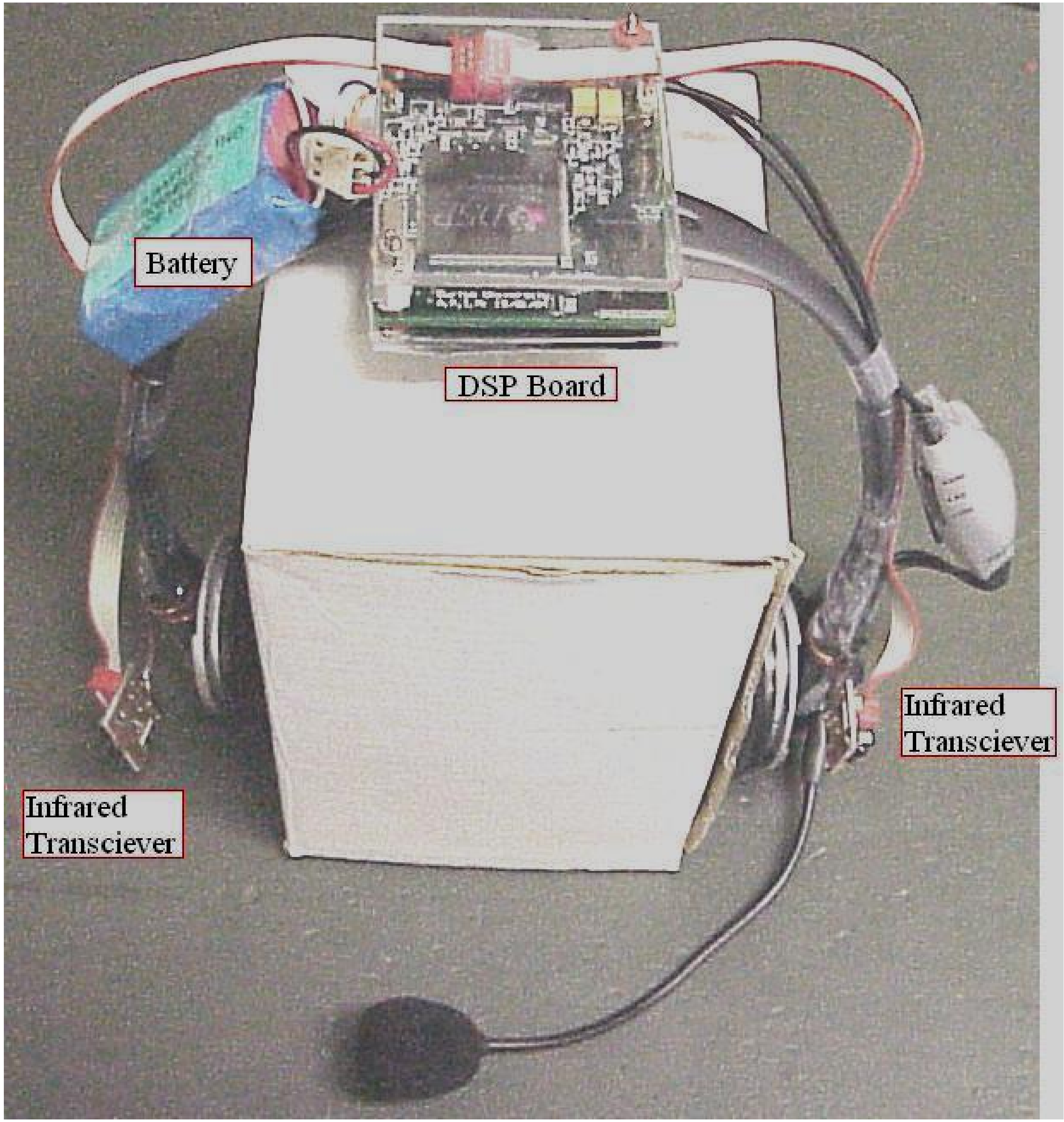


III.g. Headset PCB - Bottom View

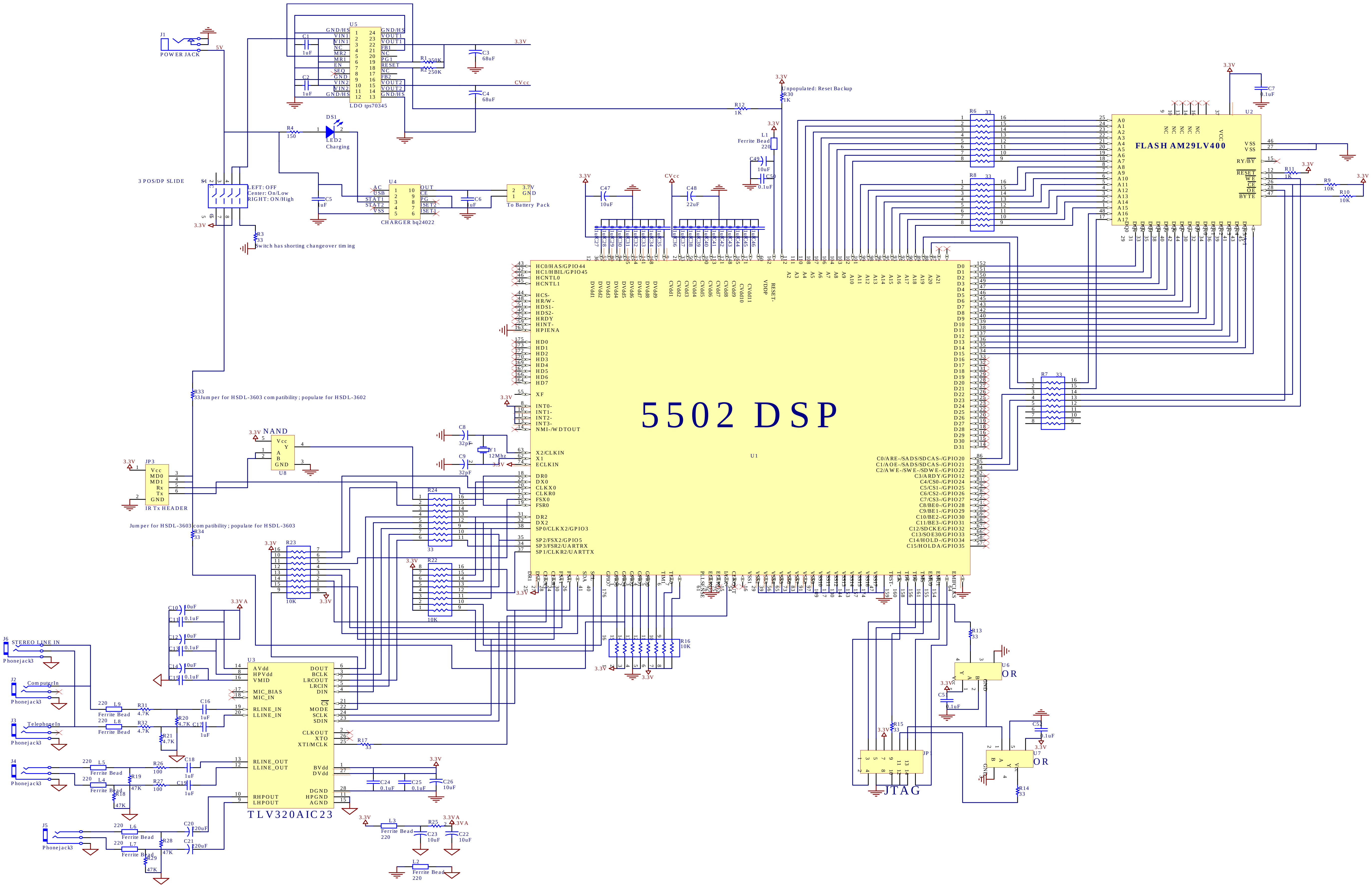


III.h. Headset Prototype #1 System

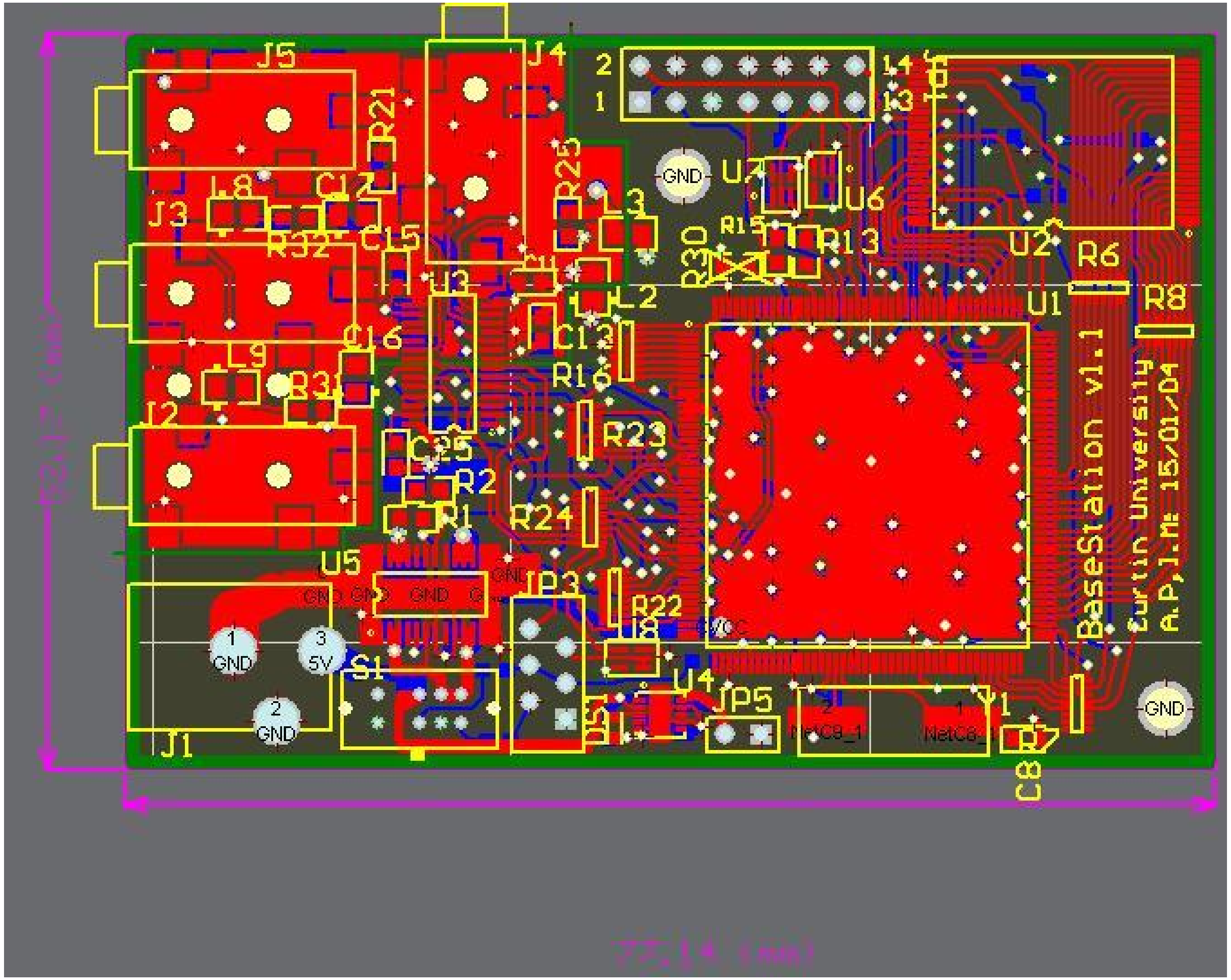



Battery
DSP Board
Infrared Transciever
Infrared Transciever

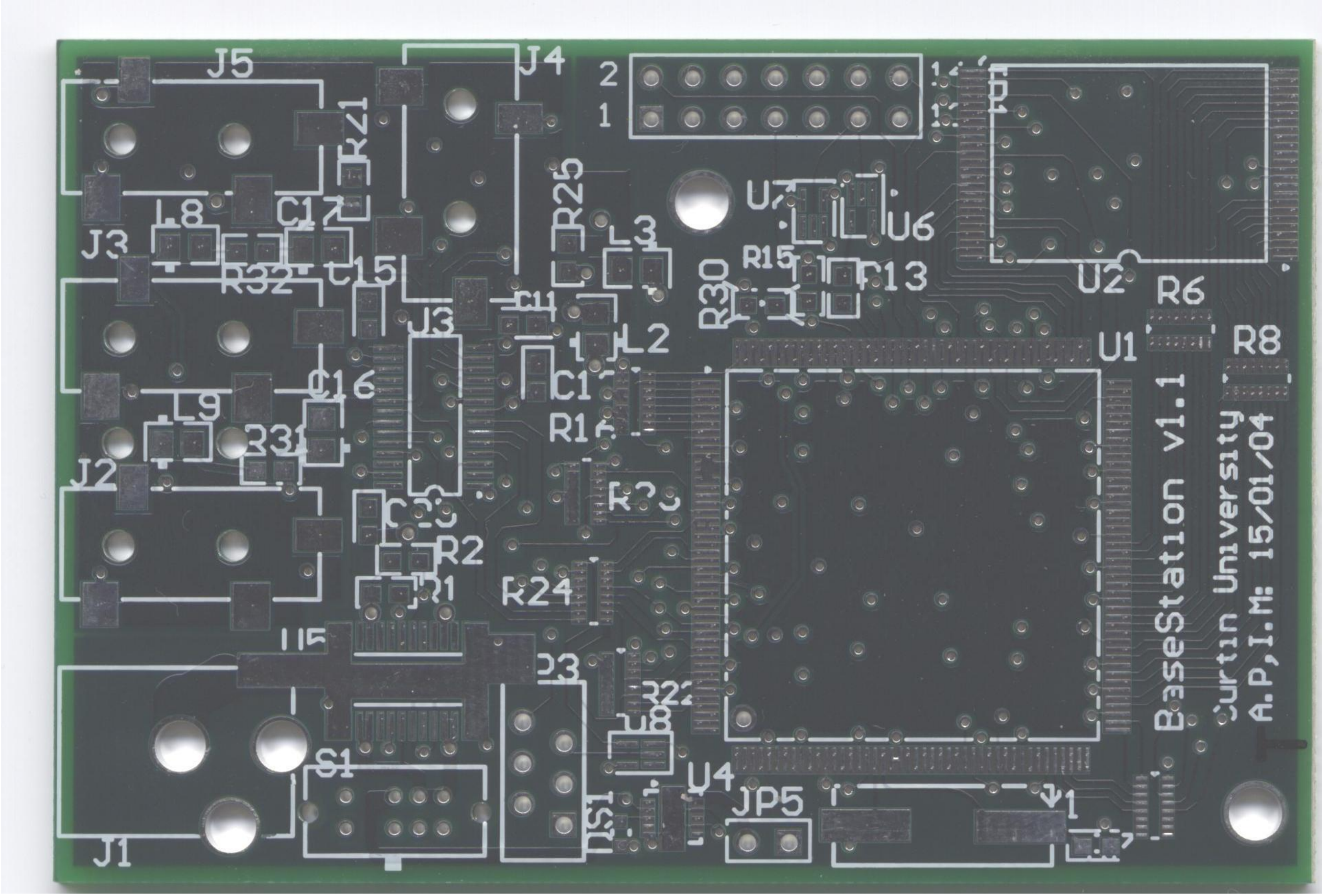
III.i. Base Station Schematic



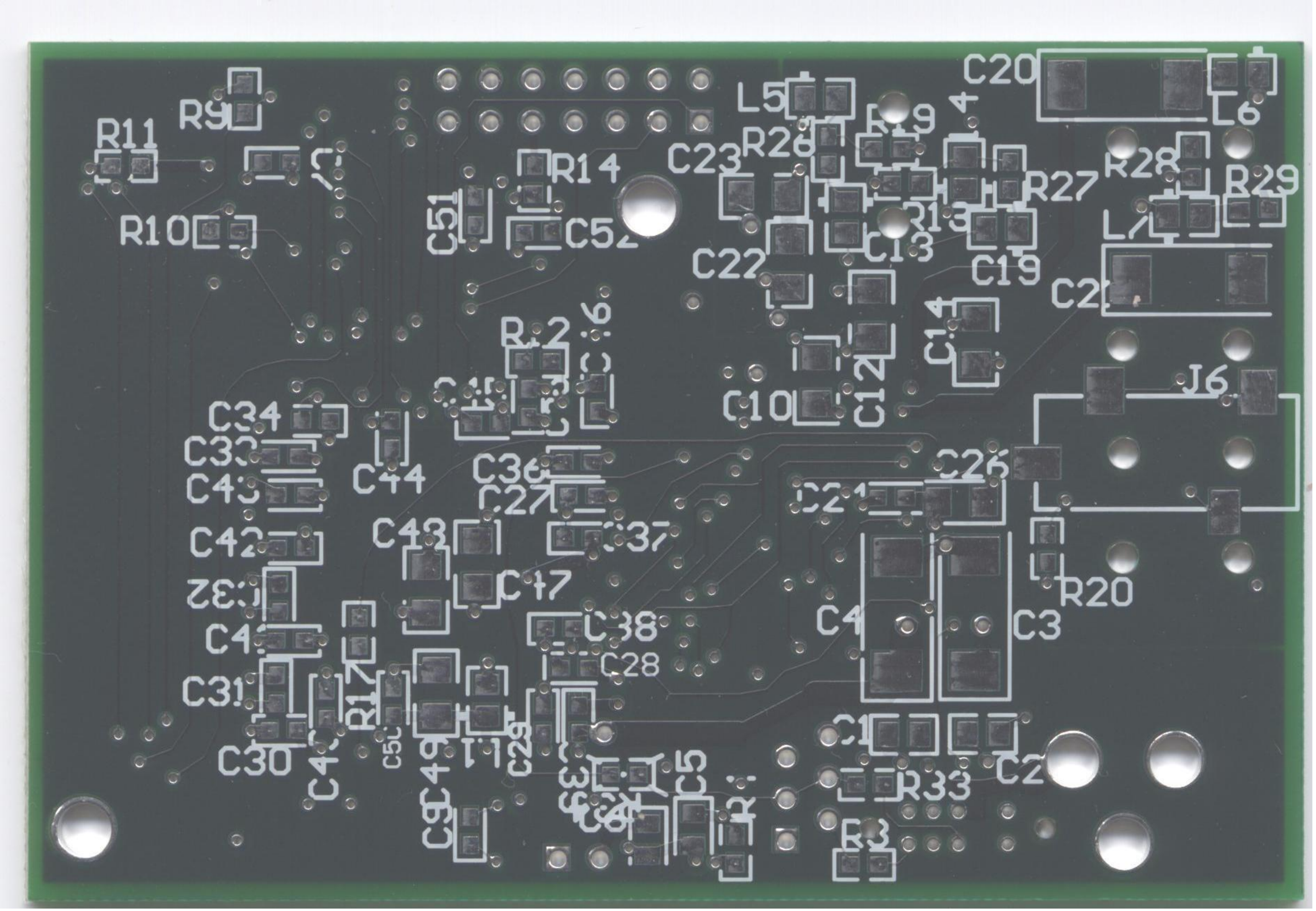
III.j. Base Station PCB Layout



III.k. Base Station Unpopulated PCB - Top View



III.l. Base Station Unpopulated PCB – Bottom View

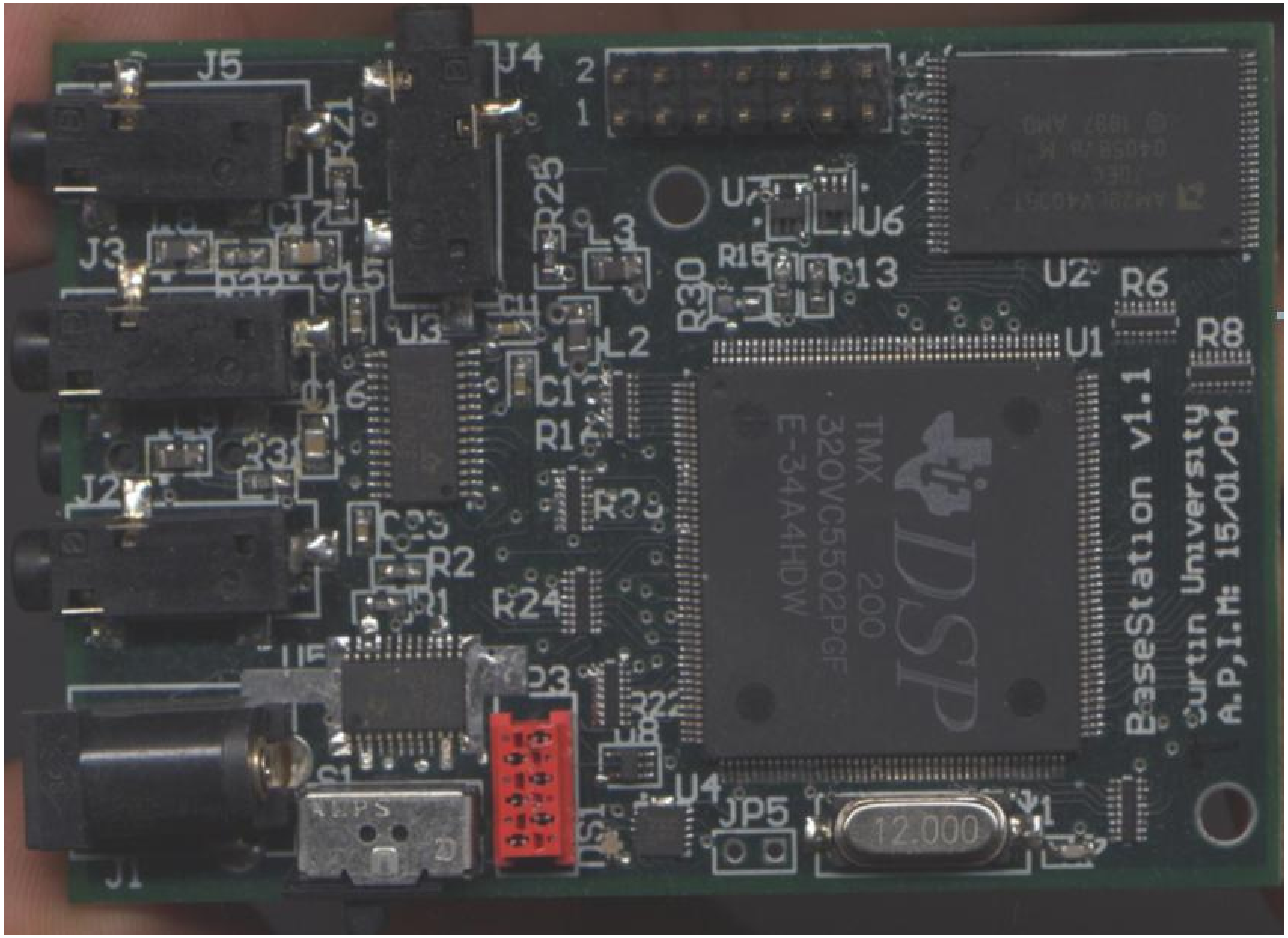


III.m. Base Station Bill of Materials

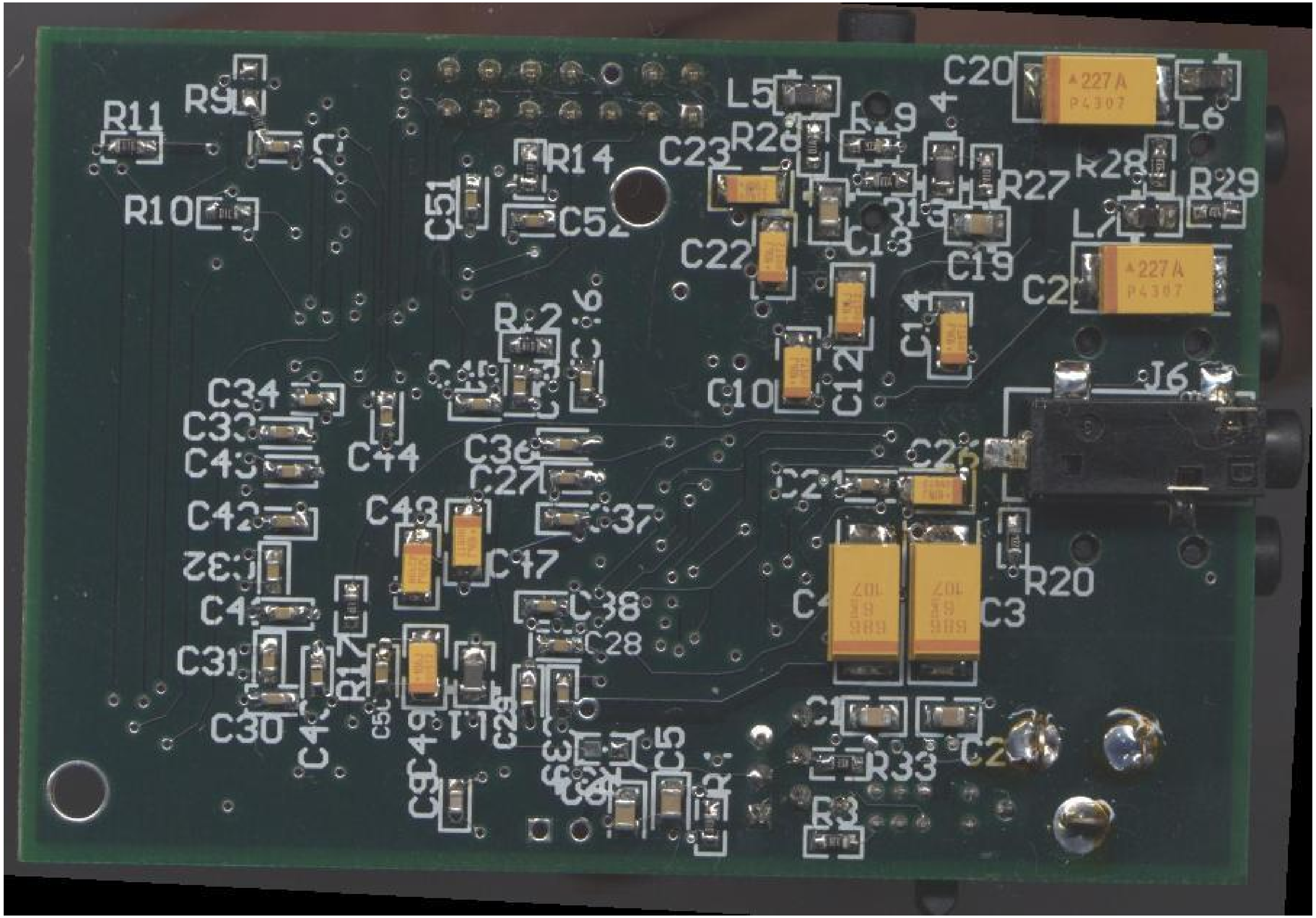
Description	Value	Designator	Footprint	Quantity	Distributor
CAP,CER,SMT 0805,1uF,6.3V,X5R,+/-10%	1uF	C1, C2, C5, C6, C16, C17, C18, C19	CC2012-0805, CC2013-0805, CR2012-0805	8	
CCAP, TANT, 68uF	68uF	C3, C4	Cap Tant Dsize	2	Farnell, #757718
CAP, CER, 0603, X7R, 0.1uF	0.01uF, 0.1uF	C7, C11, C13, C15, C24, C25, C27, C28, C29, C30, C31, C32, C33, C34, C35, C36, C37, C38, C39, C40, C41, C42, C43, C44, C45, C46, C50, C51, C52	CC1608-0603	29	
CAP, CER, 0603, 32pF	32pF	C8, C9	CC1608-0603	2	
CAP,TANT, 1206, 10uF	10uF	C10, C12, C14, C22, C23, C26, C47, C49	C3216-1206, CC3216-1206	8	
CAP,TANT,SMT,220uF,10V	220uF	C20, C21	Cap Tant Dsize	2	
CAP,TANT, 1206, 22uF	22uF	C48	C3216-1206	1	
Typical RED, GREEN, YELLOW, AMBER GaAs LED		DS1	LED HSMD-C190	1	Newark : # 98B5046
2-Conductor Jack with 1 break contact, 2.1mm		J1	PowerJack 2.1mm/2.5mm	1	Farnell: #224-959
3-Conductor Jack		J2, J3, J4, J5	StereoJack	4	Mouser: #161-4034
Header, 7-Pin, Dual row		JP1	JTAG	1	
Micro-Match Header, 6-Pin		JP3	MicroMatch 6-way	1	
Header, 2-Pin		JP5	HDR1X2	1	
FERRITE BEAD,SMT 0805,220 OHMS	220	L1, L2, L3, L4, L5, L6, L7, L8, L9	CC2012-0805	9	
Resistor	250K	R1, R2	CR1608-0603	2	
RES,SMT 0603,33 OHM,5%,1/16 WATT	33, 150	R3, R4, R13, R14, R15, R17, R33, R34	CR1608-0603	8	
Resistor Array-8, 33 OHM	33	R6, R7, R8, R24	new r8 pack	4	
RES,SMT 0603,1K OHM,5%,1/16 WATT	1K, 10K	R9, R11, R12, R30	CR1608-0603	4	
RES,SMT 0603,10K OHM,5%,1/16 WATT	10K	R10	CR1608-0603	1	
Resistor Array-8, 10K OHM	10K	R16, R22, R23	new r8 pack	3	
RES,SMT 0603,47K OHM,5%,1/16 WATT	47K	R18, R19, R28, R29	CR1608-0603	4	
RES,SMT 0603,4.7K OHM,5%,1/16 WATT	4.7K, 100	R20, R21, R26, R27, R31, R32	CR1608-0603	6	
RES,SMT 0603,2 OHM	2	R25	CR1608-0603	1	
DIP Switch		S1	Slide Switch SSAC120200	1	Mouser: # 688-SSAC120200

Description	Value	Designator	Footprint	Quantity	Distributor
5502 DSP		U1	F-QFP24x24-G176/N	1	TI
IC, AMD Flash Memory, AM29LV400		U2	TSSO12x20-G48/P.5	1	AMD
IC, CODEC TLV320AIC23		U3	TSSO10x6-G28	1	TI
Header, 5-Pin, Dual row		U4	Charger BQ-24022	1	
Header, 12-Pin, Dual row		U5	LDO tps70345	1	
IC,SO5,SINGLE 2-INPUT POSITIVE-OR GATE SN74LVC1G32DCKR		U6, U7	SO-G5/P.65	2	TI
IC,SO5,SINGLE 2-INPUT NAND GATE SN74LVC1G32DCKR		U8	SO-G5/P.65	1	TI
Crystal Oscillator		Y1	NLS7 CRYSTAL	1	

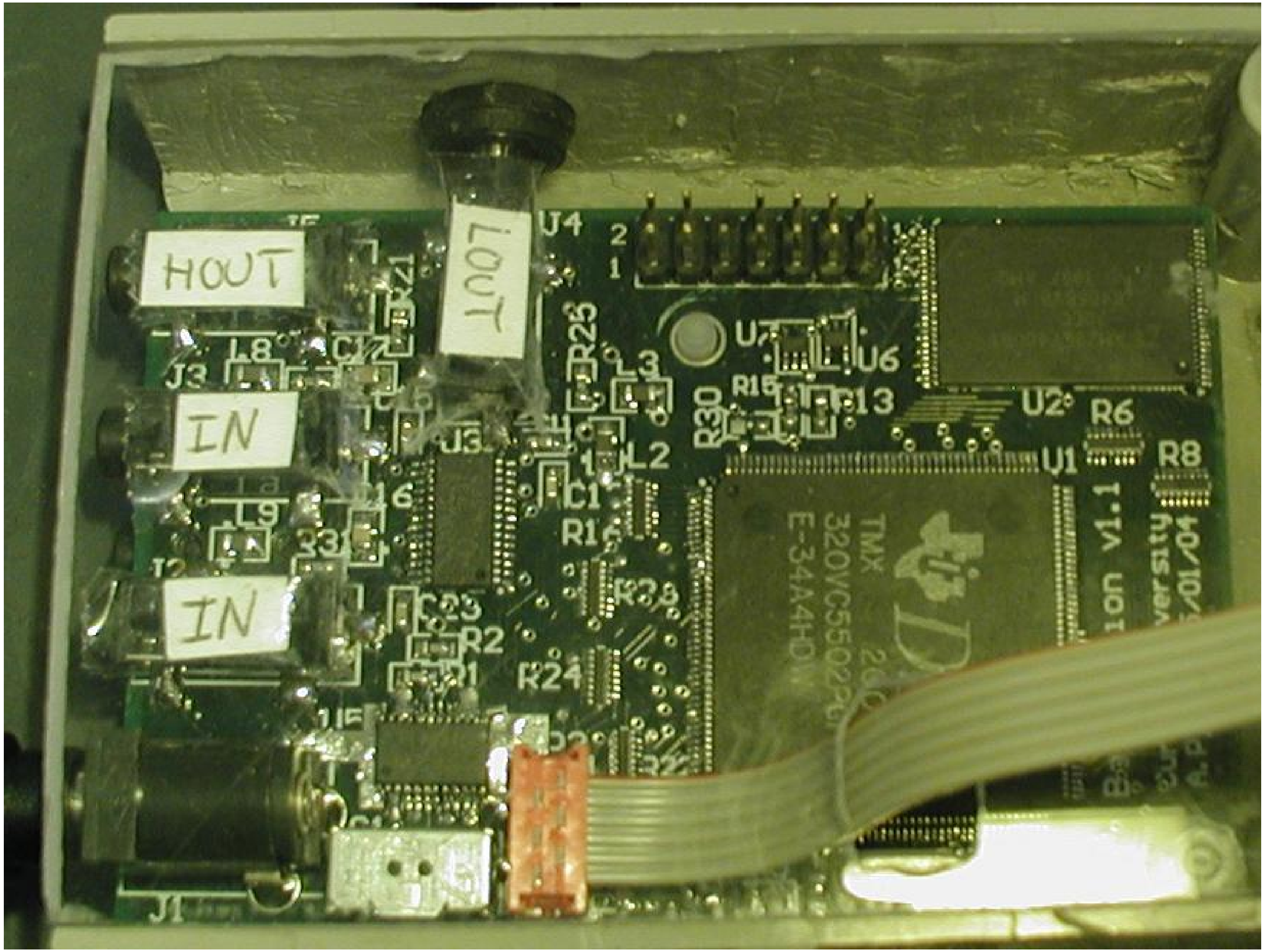
III.n. Base Station PCB - Top View



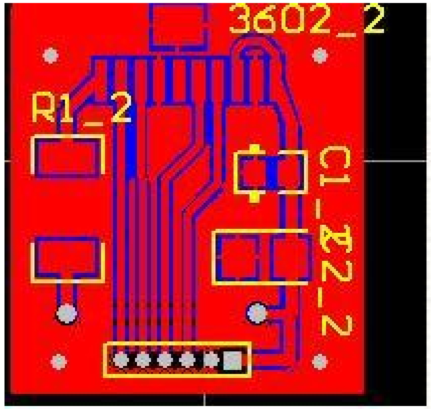
III.o. Base Station PCB - Bottom View

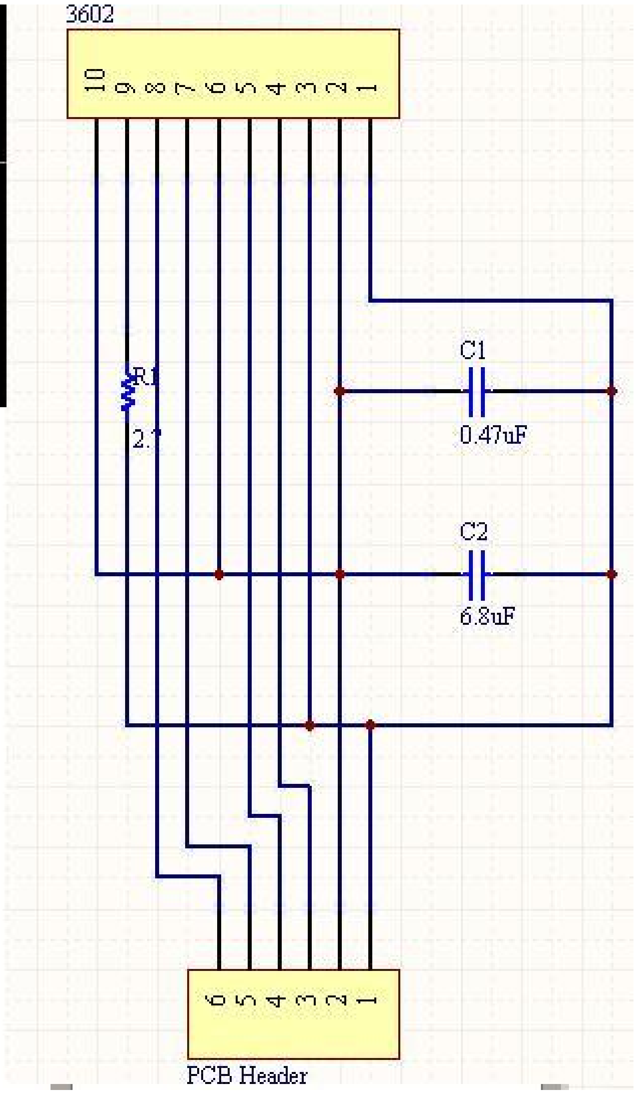


III.p. Base Station Prototype System

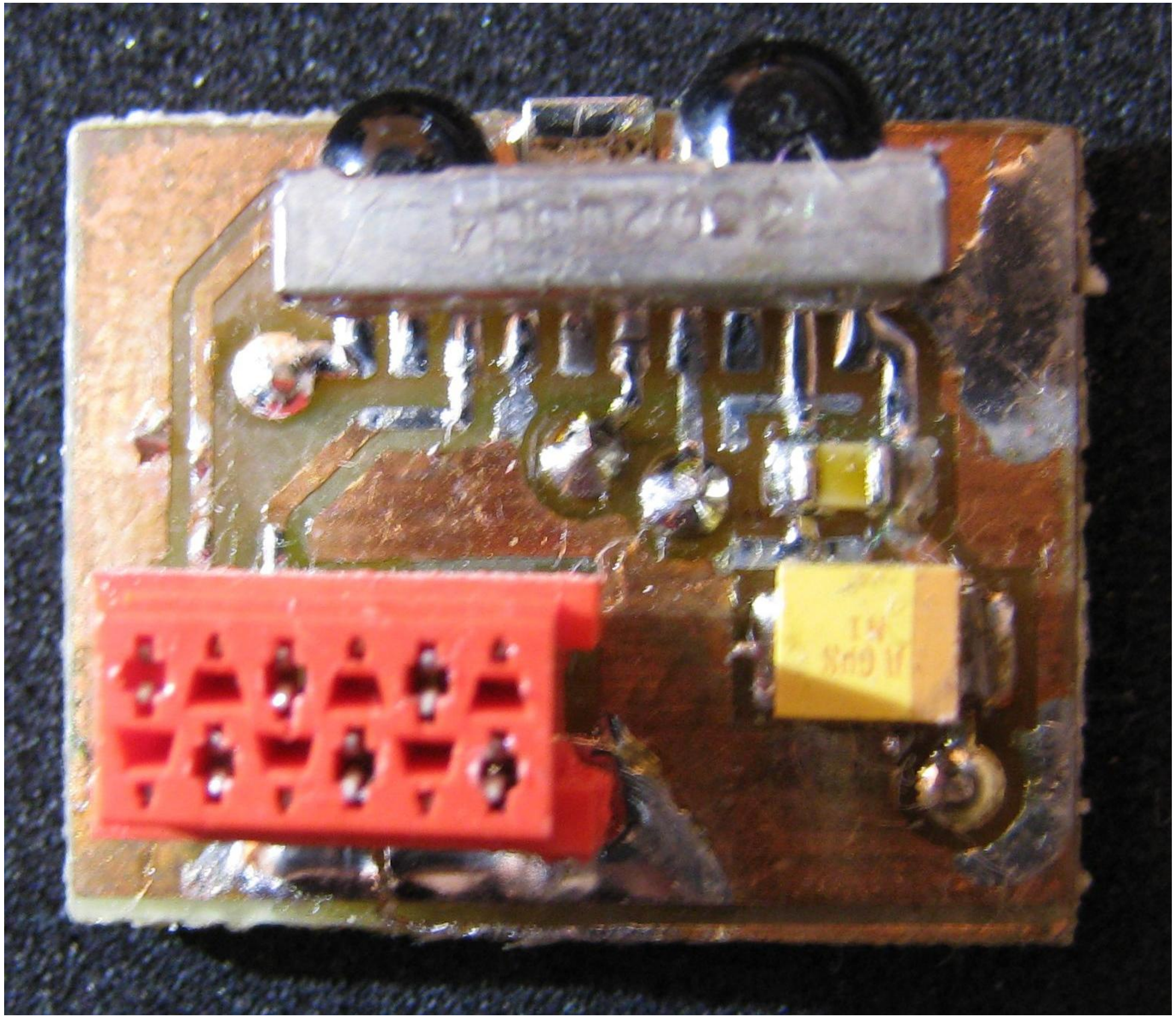


III.q. Infrared Transceiver Schematic and PCB Layout

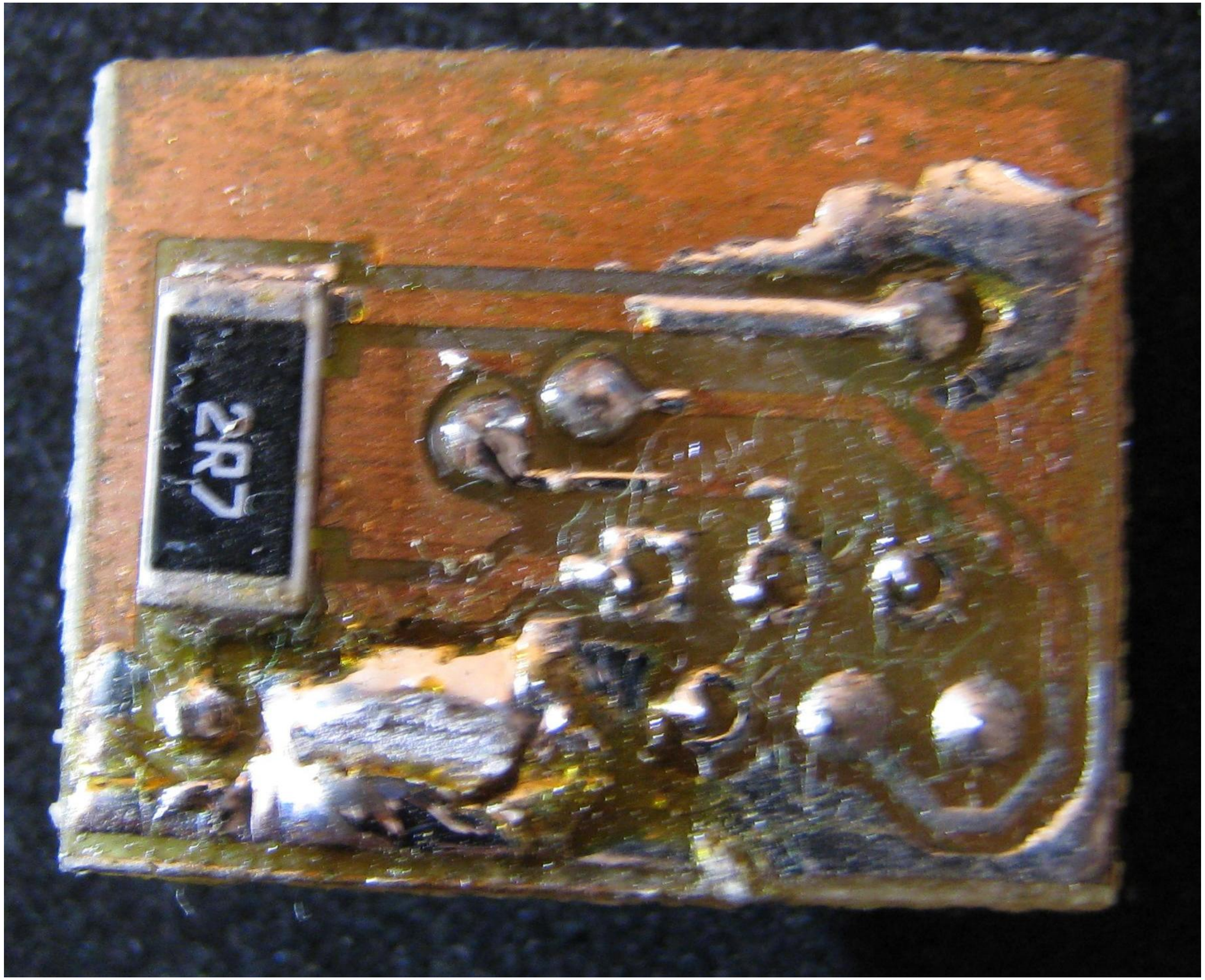




III.r. Infrared Transceiver Board – Top View

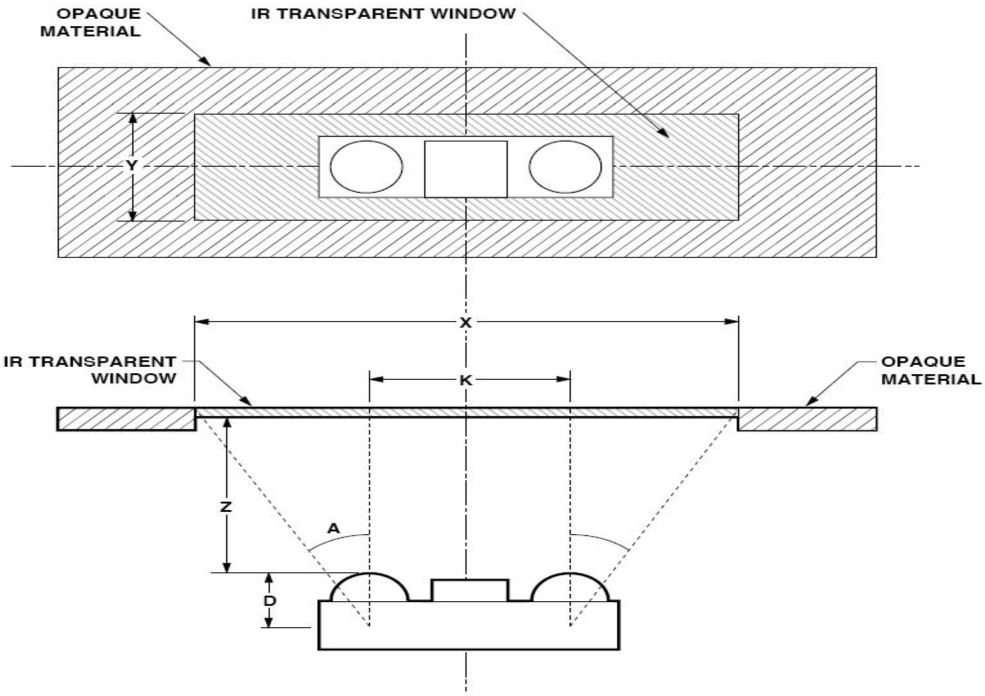


III.s. Infrared Transceiver Board – Bottom View



Appendix IV. IR Transceiver Enclosure Design

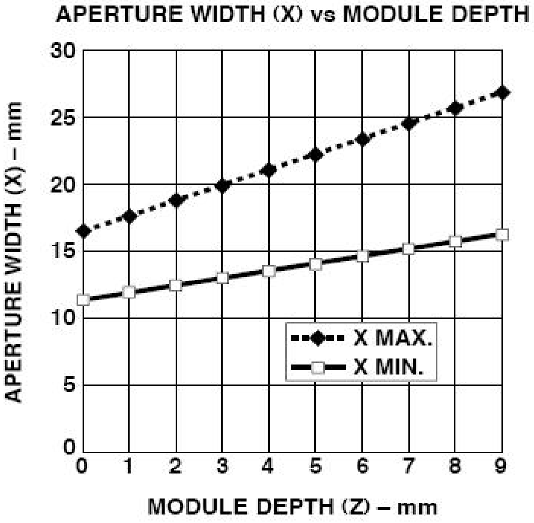
Reference source; Agilent, 2003a.

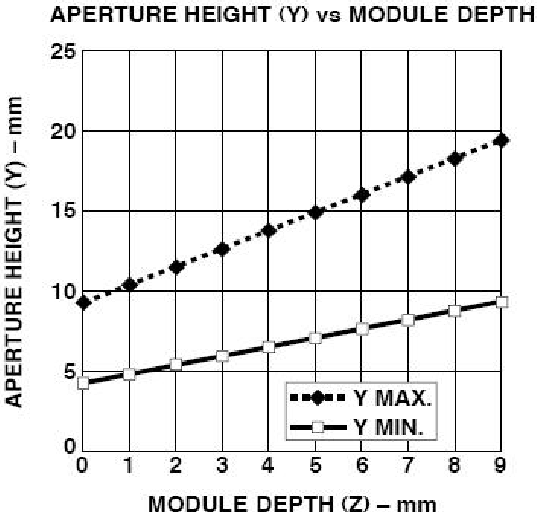


K = 7.08mm.

X = K + 2*(Z+D)*tanA

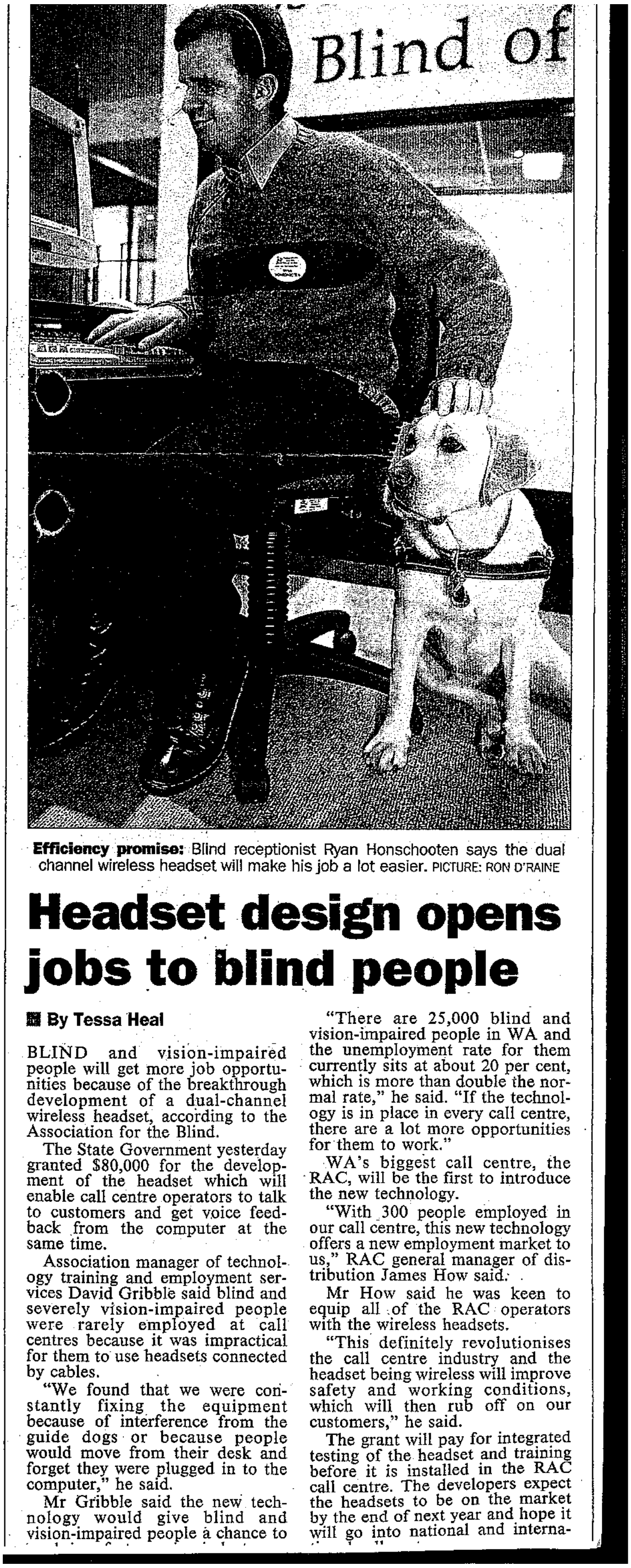
Y = 2*(Z+D)*tanA





The minimum size corresponds to a cone angle of 30º and the maximum size corresponds to a cone angle of 60º.

Appendix V. Wireless Headset Justification Article



Efficiency promise: Blind receptionist Ryan Honschooten says the dual channel wireless headset will make his job a lot easier. PICTURE: RON D'RAINE

Headset design opens jobs to blind people

By Tessa Heal

BLIND and vision-impaired people will get more job opportunities because of the breakthrough development of a dual-channel wireless headset, according to the Association for the Blind.

The State Government yesterday granted $80,000 for the development of the headset which will enable call centre operators to talk to customers and get voice feedback from the computer at the same time.

Association manager of technology training and employment services David Gribble said blind and severely vision-impaired people were rarely employed at call centres because it was impractical for them to use headsets connected by cables.

"We found that we were constantly fixing the equipment because of interference from the guide dogs or because people would move from their desk and forget they were plugged in to the computer," he said.

Mr Gribble said the new technology would give blind and vision-impaired people a chance to

"There are 25,000 blind and vision-impaired people in WA and the unemployment rate for them currently sits at about 20 per cent, which is more than double the normal rate," he said. "If the technology is in place in every call centre, there are a lot more opportunities for them to work."

WA's biggest call centre, the RAC, will be the first to introduce the new technology.

"With 300 people employed in our call centre, this new technology offers a new employment market to us," RAC general manager of distribution James How said.

Mr How said he was keen to equip all of the RAC operators with the wireless headsets.

"This definitely revolutionises the call centre industry and the headset being wireless will improve safety and working conditions, which will then rub off on our customers," he said.

The grant will pay for integrated testing of the headset and training before it is installed in the RAC call centre. The developers expect the headsets to be on the market by the end of next year and hope it will go into national and interna-

(The West Australian 2002)

Appendix VI. Contents of Accompanying DVD

- Thesis References: For local access and archiving of thesis reference documentation, especially obsolete internet URLs.
- DSP Source Code: Developed in Texas Instruments Code Composer Studio IDE v2.1.
    - Compatible for Windows 2000, Windows XP operating systems.
- PCB Designs: Developed in Protel DXP Electronic Design Package
    - Compatible for Windows 2000, Windows XP operating systems.
- Datasheets and Development Notes: For local access and archiving of relevant datasheet and research development documentation.