A LOW POWER HF COMMUNICATION SYSTEM

A thesis submitted to The University of Manchester for the degree of Doctor of Philosophy in the faculty of engineering and physical sciences

2011

John Martin Wilson

School of Electrical and Electronic Engineering

Contents

Appendices

Appendix A: Conference papers
Appendix B: Posters
Appendix C: GNU Radio source code
Appendix D: Remote unit firmware
Appendix E: Remote unit PCB files
Appendix F: Presentations
Appendix G: Progress reports
Appendix H: Buried antenna test results
Appendix I: Matlab source code

The appendices and an electronic copy of the thesis can be found on the attached CD-ROM

List of Figures

List of Tables

List of Abbreviations

8PSK	8 level phase shift keying
AAC	Advanced audio coding
ACK	Acknowledge
ADC	Analogue to digital convertor
AMTOR	Amateur teleprinting over radio
AP	A priori probabilities
API	Application programming interface
APP	A posteriori probabilities
ARM	Advanced RISC machine
ARQ	Automatic repeat request
ARRL	American radio relay league
ATU	Antenna tuning unit
AWGN	Additive white Gaussian noise
BCJR	Bahl, Cocke, Jelinek and Raviv's (maximum likelihood sequence estimation algorithm)
BER	Bit error rate
BLOS	Beyond line of sight
BP	Belief propagation
bps	Bits per second
BPSK	Binary phase shift keying
C-OFDM	Coded orthogonal frequency division multiplexing
CCIR	Comite consultatif international pour la radio
CE-OFDM	Constant envelope orthogonal frequency division multiplexing

CMOS	Complementary metal oxide semiconductor
CRC	Cyclic redundancy check
CUDA	Compute unified device architecture
DAC	Digital to analogue convertor
DAMSON	HF Doppler and multipath sounding network
DB-HARQ	Distribution based hybrid automatic repeat request
dBd	Decibels referred to a dipole
dBi	Decibels referred to an isotropic antenna
dBm	Decibels referred to 1 mW
DC	Direct current
DDS	Direct digital synthesis
DFE	Decision feedback equaliser
DPSK	Differential phase shift keying
DQPSK	Differential quadrature phase shift keying
DRM	Digital radio mondiale
DSP	Digital signal processor/processing
DVI	Digital video interface
EM	Electromagnetic
EM	Expectation minimisation
EZNEC	Antenna software by W7EL
FAX-RTTY	Facsimile radio teletype
FEC	Forward error correction
FFT	Fast Fourier transform
FIFO	First in first out (buffer)
FIR	Finite impulse response
FPGA	Field programmable gate array
FSR	Feedback shift register
G-TOR	Golay teleprinting over radio
GNU	Gnu’s not UNIX

GPIO	General purpose input/output
GUI	Graphical user interface
HARQ	Hybrid automatic repeat request
HD	High definition
HF	High frequency
HPC	High performance computing
IC	Integrated circuit
IEEE	Institute of electrical and electronic engineers
IFFT	Inverse fast Fourier transform
IONCAP	Ionospheric communications analysis and prediction program
ISI	Inter-symbol interference
ISR	Interrupt service routine
ITU	International telecommunication union
ITU-R	ITU radiocommunication sector
JTAG	Joint test action group
LDPC	Low density parity check
LLR	Log likelihood ratio
LNA	Low noise amplifier
LO	Local oscillator
LT	Luby technique (a.k.a fountain coding)
LUF	Lowest usable frequency
MAP	Maximum a-posteriori
MFSK	Multiple frequency shift keying
MLCM	Multi-level coded modulation
MLSE	Maximum likelihood sequence estimation
MMIC	Monolithic microwave integrated circuit
MPEG	Motion pictures experts group
MT	Multi tone
MUF	Maximum usable frequency

NAK	Negative-acknowledge
NATO	North Atlantic treaty organization
NVIS	Near vertical incidence skywave
OFDM	Orthogonal frequency division multiplexing
OMAP	Open multimedia application platform
OQPSK	Offset quadrature phase shift keying
PA	Power amplifier
PACTOR	Packet based amateur teleprinting over radio
PAPR	Peak to average power ratio
PC	Personal computer
PCB	Printed circuit board
PIC	Brand of microcontroller by Microchip
PN	Pseudorandom
PSK	Phase shift keying
QAM	Quadrature amplitude modulation
QC-LDPC	Quasi-cyclic low density parity check
QI	Quadrature/in-phase (signalling)
QPSK	Quadrature phase shift keying
QRP	Low power ham radio operation
RAM	Random access memory
RB-HARQ	Reliability based hybrid automatic repeat request
REC533	Channel propagation software based on ITU-R recommendation F.533
RF	Radio frequency
RLC	Resistance, inductance, capacitance (filter)
RRC	Root raised cosine (filter)
RSE-HARQ	Random sum of elements hybrid automatic repeat request
RTTY	Radio teletype
SBC	Single board computer
SCS	Spezielle Communications Systeme

SDR	Software defined radio
SFDR	Spurious free dynamic range
SFF-SDR	Small form factor software defined radio
SIMD	Single instruction multiple action
SITOR	Simplex teletype over radio
SNR	Signal to noise ratio
SPI	Serial peripheral interface
STANAG	Standard agreement (NATO standard)
SWIG	Simplified wrapper interface generator
SWR	Standing wave ratio
TQFP	Thin quad flat pack
UART	Universal asynchronous receiver/transmitter
UK	United Kingdom
USB	Universal serial bus
USD	United states dollars
USRP	Universal software radio peripheral
VGA	Video graphics array
VOACAP	Voice of America coverage analysis program
VSWR	Voltage standing wave ratio

Abstract

The HF band of radio frequencies, from 3-30 MHz, is unique in its property that it is refracted by the ionosphere. This property allows long distance radio telecommunications around the world without requiring infrastructure. High frequency (HF) communication has been largely superseded by satellite and cellular technologies for day-to-day communications, due to the tight bandwidth constraints and technical difficulties inherent in using it. However there is still a need for HF communications devices where existing infrastructure is not available, such as in remote or polar locations, or in emergency situations due to natural disasters.

This research is aimed at the development of an asymmetric HF communications link, with a battery-powered remote unit that transmits a small amount of data to a mains-powered base station. New technologies are identified and evaluated for use in the link, with the aim of reducing the power requirements of the remote unit.

Error correction techniques are investigated. Low-density parity check (LDPC) codes, which are powerful codes used for forward error correction, are suggested for use in the link. Quasi-cyclic LDPC codes allow the low-power transmitter unit to use a computationally simple encoder based on feedback shift registers for generating the LDPC block codes cheaply. Semi-blind LDPC turbo equalisation is a powerful technique that can be used at the base station which utilises the structure of the LDPC code to encode the data stream. This equalises a received signal with a minimal amount of training data required, reducing the duty cycle of the remote unit. Hybrid automatic repeat request (HARQ) techniques are also investigated, which increase the throughput of a link when data repeats are required. A novel HARQ technique was created and proven to increase throughput in links with noise.

As the proposed system may be deployed in remote locations, or locations where it might be difficult or undesirable to erect a proper HF antenna, a selection of buried antennas are characterised.

A design for a remote unit is suggested. This unit was manufactured and used to test the capability of inexpensive, low power hardware to implement the proposed remote unit algorithms.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright statement

Acknowledgement

I would like to take this opportunity to thank my supervisor, Peter Green, for his patience, good humour and continuing support over the last four years.

I would also like to thank my corporate liaison, Dr. Ben Allen, for his professional assistance, guidance and thorough proof-reading provided throughout my Ph.D.

Chapter 1

Introduction

The problem of long distance communication is one ubiquitous to all of human civilisation, from ancient smoke signalling to modern radio and wireless infrastructure. Communication today is conducted mostly through a well established wired, wireless and satellite infrastructure which allows near instantaneous communication between any countries in the world. This infrastructure includes wired and fibre-optic telephone and cable connections to buildings, cellular communications infrastructure, radio and television broadcast services and the internet. Ad-hoc radio networks are used where reliance on established infrastructure is undesirable or impossible, such as for the emergency services or the military.

1.1 Project Description

The aim of this project is to design a digital communications system suitable for long range, over-the-horizon transmission of information from a remote unit to a base station, where data transfer will predominantly take place from the remote unit to the base station. The system must not rely on any external infrastructure such as satellites or mobile communications infrastructure to aid it in transferring data. This necessitates operating in the HF (high frequency) band of radio frequencies, from 3-30 MHz, because this band of frequencies refracts off the ionosphere allowing over the horizon radio communications. Chapter 2 provides a brief discussion of the HF channel and speculative channel conditions likely to be encountered.

The communications link will consist of a remote unit and a base station. The remote

unit will transmit data from a remote location to a base station with a range in the order of hundreds of miles. The remote unit will be a small, battery powered device which will be opportunistically placed in a potentially hostile environment. As such, the antenna is likely to be rudimentary and of poor design, construction and location. The receiver station will be a larger unit, mains powered and probably located within a building. There are as such few limitations to its design and the receiver will be assumed to be based on powerful hardware or software radio with a high gain, directional antenna. The remote unit, base station and possible antennas are discussed in more detail in chapters 10, 11 and 9 respectively.

Due to the nature of the transmitter unit, being battery operated and with a poor antenna, the system must be designed to work in very low signal-to-noise ratio conditions. This necessitates careful consideration of error detection and correction techniques. Forward error correction (FEC) will be used to encode redundancy into the transmitted data to reduce the bit-error rate (BER) of the decoded data at the receiver. This will be paired with automatic repeat-request (ARQ), in which resends of any erroneously received data from the receiver to the transmitter are requested by the receiver. Chapter 7 discusses modern FEC and ARQ techniques suitable for the system.

1.1.1 Project motivations

Most of the recent product development of HF-band radio systems has focussed upon providing wide bandwidth systems capable of supporting high data rates (in the region of 9,600-19,200 bps). There is also currently a lot of interest in the establishment of secure communications networks, monitoring of the HF spectrum, and performing direction finding on detected signals for defence and national security establishments. Most of the modern standards, such as the forthcoming United States military standard MIL-STD-188-110C and the Digital Radio Mondiale [6] reflect this.

Due to the practical difficulties associated with using HF communication links, together with the well established worldwide communications infrastructure, less emphasis has been placed on developing smaller, portable and power efficient systems. Such devices may, however, prove useful for a range of applications where the established communications infrastructure is either difficult to use or unavailable. Remote monitoring stations, for example meteorological or seismic detection systems may be located out of cell phone range. In the north and south poles, it

becomes difficult to establish communication links with satellites, and impossible to do so at latitudes further north or south than 82°. Such applications are unlikely to require high bandwidth bi-directional links. However they are unlikely to be mains powered therefore power efficiency is of high priority.

As mentioned in the previous paragraph, the applications for which a low-powered link would be useful may typically be asymmetric, with a remote, unmanned unit sending data back to a base station. The amount of data sent is unlikely to be a large amount, perhaps simple readings from a variety of sensors that are compiled and sent in a burst every few days. The base station would most likely be manned and, as such, have access to mains power supplies. This project therefore focussed upon the development of such a system. The work performed was mainly research into techniques that allow the implementation of a remote unit that transmits data to a base station, keeping the remote unit's power requirements to a minimum by using more complicated, processor intensive algorithms at the base station. The remote unit hardware is based around a low power DSP (digital signal processor) microcontroller. The design of the remote unit is important in the respect that over-complicating its design is likely to increase the power requirements. The base station was based around a software radio platform. This allows a high-performance, scalable solution whereby the computational and signal processing power of the base station can be increased by using a more powerful computer system, or even a network of computers, to perform processing of the received data after receiving the transmission.

1.1.2 System Block Diagram

Figure 1.1 shows a system block diagram for the proposed communications link. Data to be sent is represented by the top left block, this is then encoded with FEC code to add redundancy to the signal. The block after this is a repeater block which will arrange for repeats of the transmitted data to be sent on request from the receiver via the feedback link. The next block on the path is modulation, which represents the transform of the string of data bits to symbols suitable for sending through the channel. The channel block is representative of all the effects of the HF channel including multipath Rayleigh fading, Doppler spread, Doppler shift and additive noise. The equaliser block represents a section of the receiver station that attempts to compensate for the channel's effects on the received signal to optimise demod-

ulation. The next block, demodulation and clock recovery, is representative of the process of demapping the channel symbols back to a string of data bits. The original data clock must be recovered from the received signal to perform this operation. FEC decoding removes the redundancy from the data signal, this block is also assumed to detect errors in the decoded signal (i.e. detect a decoding failure) in which case a repeat request is issued via the feedback link to the transmitter. The feedback link is depicted separate from the channel because, although the physical conditions are likely to be similar, the modulation techniques are likely to be different, due to the low bandwidth nature of the feedback link (a minimum of one signal is required to acknowledge valid data decoded at the receiver).

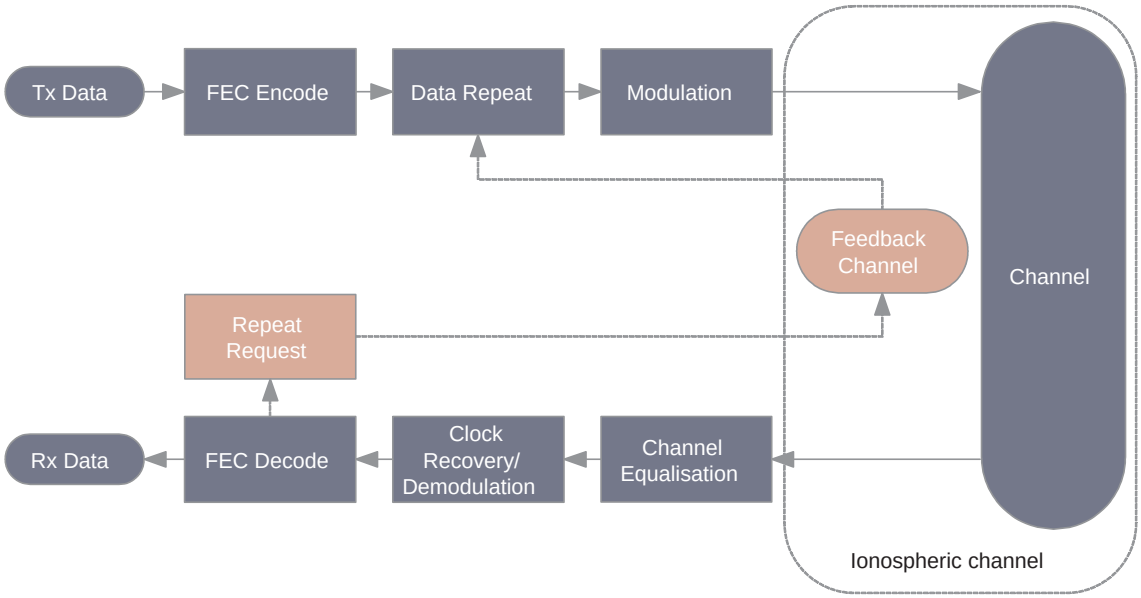


Figure 1.1: Block diagram of the proposed system

1.2 Synopsis of original work

A number of techniques suitable for use in a modern HF modem were developed and implemented using the GNU (GNU's not Unix - a collaboration of open source developers) Radio software radio application programming interface (API) and the Ettus technologies universal software radio peripheral. They were developed to implement the base station test unit for the system. It is intended to release the source code for these to the public domain in the future. The development of the base station is described in chapter 11.

The GNU Radio platform was used to realise a software radio based Watterson

channel model which is capable of operating at RF frequencies. The design and implementation of this system is discussed in section 2.7.

A novel type II hybrid-ARQ (HARQ) technique has been devised which increases the throughput of a system over that expected from conventional chase combining of random bits which requiring the same simple acknowledge/negative-acknowledge (ACK/NAK) capable feedback link. This is discussed in section 7.10 and a performance evaluation is given in section 12.5.

A software radio system based capable of receiving and performing channel equalisation on low density parity check (LDPC) coded data corrupted by AWGN (additive white Gaussian noise) and the ITU-R (international telecommunication union - radiocommunication sector) recommended HF channels using semi-blind turbo equalisation was developed. The technique is described in section 8.3. A performance evaluation was performed and the results are presented in section 12.3.

A number of buried antennas were designed and there performance was evaluated. The design and evaluation results are presented and discussed in chapter 9.

A printed circuit board (PCB) was developed and manufactured to implement the remote unit of the proposed system. This was intended as a proof of concept to prove that a serial tone modem that encodes data using the modern LDPC forward error correction technique could be implemented on a low power fixed point DSP controller. The design of this board is described in chapter 10.

Chapter 2

The HF channel

The frequency band between 3-30 MHz, known as the high frequency, or HF band has interesting properties that allow beyond line of sight (BLOS) communications. In this frequency band, the radio signals will 'reflect' (or, more accurately, refract) from an electrically charged layer of the atmosphere known as the ionosphere. Below this band of frequencies radio signals tend not to be reflected sufficiently to reflect back to earth, and above these frequencies signals will tend to be absorbed more readily by the atmosphere, attenuating the signals too much for effective long-distance communications.

2.1 Ionospheric Reflection

The ionosphere is the upper region of the Earth's atmosphere, starting at an altitude of approximately 80 km. In this region the atmosphere is ionised strongly by solar radiation and it is so sparse that electrons ejected during the ionisation process can exist freely for a substantial length of time before they are reabsorbed by an ion. These properties result in a high free electron density in this region, up to approximately 10^{11} electrons/m^3 [7]. These areas of high electron density have the effect of reflecting (or more accurately, refracting) incident radio signals back to earth, which allows for long distance, over the horizon radio communications in the frequency band associated with the reflections. The frequency band used for ionospheric communications through a given channel is a function of the electron density of the ionospheric layer upon which it is reflected and the incident angle of the sig-

nal. The electron density is dependent on the amount of ionisation taking place from irradiation by the Sun; seasonal changes and the sunspot number, which is closely correlated with solar activity, are as such also factors influencing the ionospheric channel.

The composition of the ionosphere, coupled with the effects of height variation on the pressure-sensitive ionisation process, manifests distinct layers of high electron density at different altitudes. These are described in the following subsections.

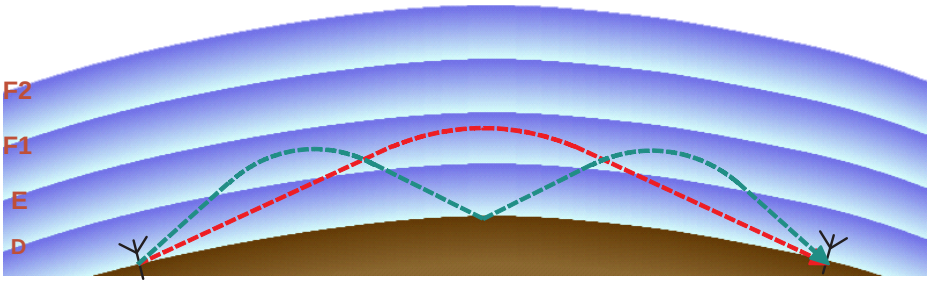


Figure 2.1: The HF channel layers

2.1.1 The D layer

The D layer is the lowest layer of the ionosphere, forming between the altitudes of 60 to 90 km [7]. It is only seen during daylight hours and mainly absorbs HF signals.

2.1.2 The E layer

The *E*-layer sits from approximately 90 to 130 km above the earth's surface and reaches its maximum ionisation when the sun is at its zenith at noon [7]. This layer discharges slower than the D layer, reaching its minimum ionisation at midnight. It reflects signals of less than 10 MHz in frequency but can contribute to absorption losses of higher frequency signals. Its location closer to the surface makes it useful for short range or near-vertical incidence skywave transmissions where the transmitter and receiver are less than 500 km apart.

Occasionally patches form in the *E*-layer with a higher ion density than the surrounding atmosphere, due to pressure effects such as wind shear. These patches

can reflect much higher frequencies. They are known as sporadic-E (E_s) and are most pronounced in the summer months.

2.1.3 The F layer

The F-layer forms at altitudes between 130 to over 500 km, where ions recombine slower than the other layers [7]. At night time it is roughly homogeneously distributed, but at daytime two distinct layers are formed; the F_1 layer lower than 250 km and the F_2 layer higher than that. The F_1 layer is more pronounced over the summer months and during periods of high solar activity. The F layer can reflect frequencies from between approximately 10 MHz to 30 MHz and is responsible for most skywave broadcasts.

2.2 Maximum and minimum usable frequencies

The changing properties of the ionosphere, including the daily and seasonal variations of the layers give rise to a definable range of frequencies within which a transmission can theoretically take place at a given signal to noise ratio (SNR). The range is defined by a maximum and minimum usable frequency, which are generally referred to in the literature as the maximum usable frequency (MUF) and lowest usable frequency (LUF). It is important that a reasonable estimate of these parameters is known before communications are attempted, and they rely on a number of factors such as geographical locations of transmitter and receiver, date and time. Programs such as VOACAP [8][9] (voice of America coverage analysis program) automatically estimate these parameters from an internal model (discussed in section 5.6.1).

2.3 Effects of the ionosphere on skywave signals

The ionosphere is not uniform and is constantly in motion. This, combined with its multiple layers, result in many different forms of distortion including Doppler shift and spread, multipath and fading.

2.3.1 Doppler shift

The properties of the ionosphere are constantly in flux; daily and seasonal variations, and changes in solar activity cause the electron density of the ionospheric layers, as well as their heights, to change dramatically. These changes result in a Doppler shift on a reflected signal as the layer shifts in altitude over the course of a transmission.

2.3.2 Doppler spread

When a signal bounces off a layer in the ionosphere, the signal seen at the receiver is an ensemble of reflections reflected from different positions around the mean point that the signal is reflected. This is due to the fact that the ionosphere is not perfectly uniform. A good analogy to this is watching a sunset reflected on the sea. The small perturbations in the water (the waves), cause the reflection of the sum to be spread out over a wider area. This process introduces Doppler spread into the signal, as the ionosphere is constantly moving (like the waves in the ocean). The *Doppler spread* of a signal is the amount by which the received components of the signal differ in frequency, this can also be described by means of a *Doppler spectrum*, which is in essence the probability density distribution of the received signal frequencies. The bandwidth of a Doppler spectrum is the Doppler spread. Doppler spread is closely related to fading, which is discussed in the next section.

2.3.3 Fading

Interference between the components of a Doppler spread signal, which will have different time of flights with respect to each other, manifests as a random phase and amplitude change of the received signal. When the received signal is in flux (for example when the signal is bounced off a moving reflector) it gives rise to *fading* [10], which is observed as random fluctuations of the received signal's phase and amplitude. Two common fading models used in communications systems analysis are *Rayleigh fading* and *Rician fading*. The former assumes that a signal received through a channel is subject to amplitude variations given by the Rayleigh distribution, this assumes a large number of scatterers (originating in the ionosphere). The latter is similar but assumes an additional strong line-of-sight signal with no

such distortion. The Rayleigh fading model is typically used for HF systems analysis and channel models such as the Watterson model [1] because the typically long distances involved cause Doppler spreading of all received signals.

Fading channels can be sub-categorised by the characteristics of the fading with respect to the signal. If the bandwidth of the signal is less than the channel's *coherence bandwidth*, i.e. the amplitude and phase change can be assumed constant through the whole bandwidth of the signal, then the term *flat fading* is applied. This might not happen in wideband signals, in which case the term *frequency selective fading* is used. If the fades happen quickly compared to the sampling frequency of the signal, such that a deep fade might begin and end within a symbol period, then this is known as *fast fading*. This is typically more difficult to compensate for than the converse, *slow fading*, which happens more slowly and has an effect over the course of many data symbols. These properties have implications for the design of the equaliser used to receive the signals.

2.3.4 Multipath distortion

In addition to the Rayleigh spreading mentioned above, another related distortion seen in the HF channel is multipath. Multipath is a phenomenon seen when a signal takes more than one path to get from the transmitter to the receiver. Different paths will, in general, be of different lengths and correspondingly impose different delays on their signals, therefore a received multipath signal is the superposition of multiple time-delayed images of the original transmitted signal. Multipath can occur in a HF context with two skywave paths, one of which refracts once off the ionosphere and the other which refracts twice or more, reflecting off the ground between. In addition to these, ground waves are also present, which result from diffraction of radio waves around the earth surface. Each of these paths will result in a different image being seen at the receiver. Multipath exists in all communication systems but in long range HF systems the distortion is particularly severe as the long distances involved result in delays in the order of milliseconds from when the first image is received to the last.

Figure 2.2 shows four different types of multipath, clockwise from the top left, these show the signal refracting from two different layers simultaneously, i.e. not all of the power is reflected at the E-layer and some passes through to be reflected by the F-layer. A similar type of multipath to this can occur when a high incidence

component of a transmitted signal is reflected in the ionosphere slower than a lower incidence component, resulting in a longer path for the former. The second diagram shows a skywave and ground wave combining at the receiver to form multipath, where the ground wave has a shorter path. The bottom right diagram shows a phenomenon known as magneto-ionic splitting whereby a signal being transmitted into the ionosphere is split by the earth's magnetic field into two oppositely polarised signals known as the ordinary and extraordinary components of the original signal. The paths of these two components can differ in length resulting in multipath. The last diagram on the bottom left is the case of multiple hops discussed above, where a transmitted signal can bounce multiple times off the ground and ionosphere.

Doppler shifts imposed upon the signals travelling through different paths by the altitude shifts of the ionosphere may be different for each path. A groundwave is likely to have a very low (or non-existent) Doppler shift, due to the signal not scattering as it would when reflecting off the ionosphere, whilst paths taking multiple hops may have much higher Doppler shifts. This is because Doppler shifts from the ionosphere at each reflection are likely to be correlated. When two paths are received with different, varying Doppler shifts, frequency selective fading can act on a part of the signal. As the Doppler shifts of the paths change relative to each other, fades can move across the bandwidth of the signal.

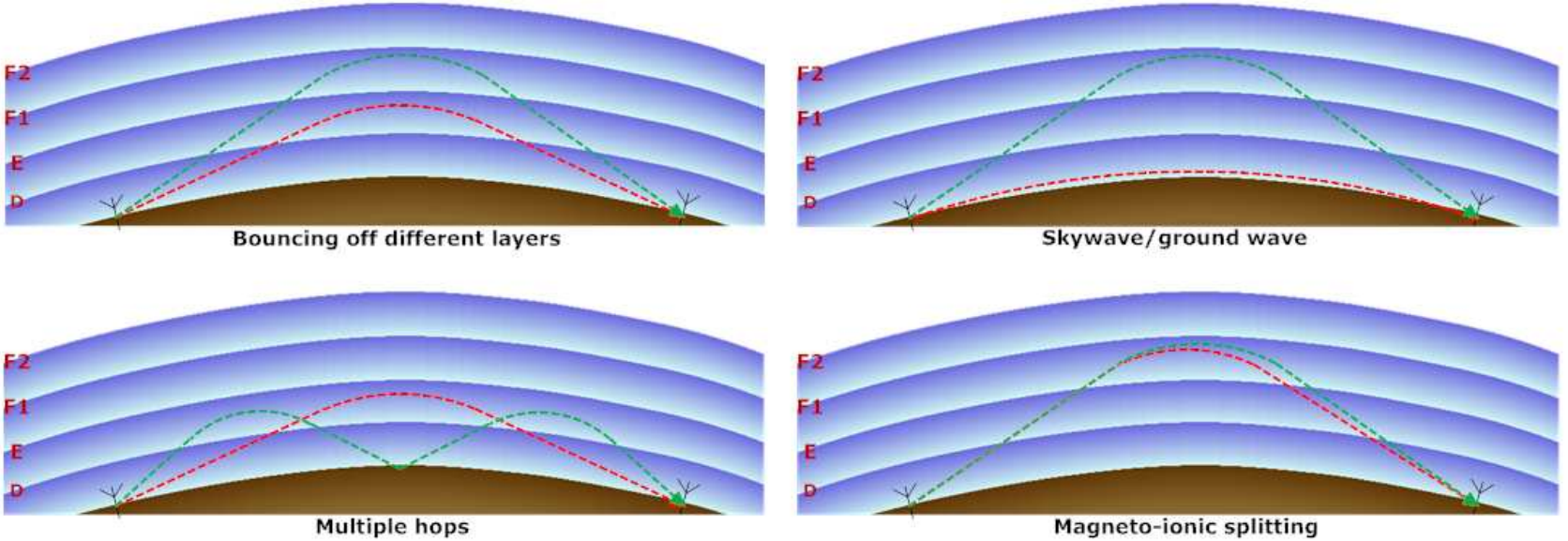


Figure 2.2: Possible paths that a HF signal can take

2.3.5 Compensating for ionospheric distortion

The Rayleigh fading and multipath distortions observed in a received signal will reduce the performance of many radio receivers as phase distortions cause symbols to be decoded incorrectly and amplitude fades cause the signal-to-noise ratio (SNR) to decrease sharply. To compensate for these effects radio receivers use channel equalisation, which implements techniques designed to correct the phase and amplitude of the received signal using prior knowledge of the signal and channel properties, which may be a intrinsic in some encoding scheme used on the signal at the transmitter, or by known training data that is sent alongside the unknown data payload of the transmission [3]. To help correct for deep amplitude fades, which cause the SNR to dip to a level so low as to make any data transmitted during that fade effectively unrecoverable, a combination of forward error correction (FEC) and interleavers can be used. The interleaver has the effect of spreading a burst of errors, resultant from a deep fade, evenly across a much longer length of the transmission. This gives the FEC mechanism a greater chance of faithfully recreating the original data.

Chapter 8 gives a description of the FEC coding and equalisation techniques which were decided upon for integration into the final system. A family of low density parity check (LDPC) codes was chosen. An LDPC encoder has the effect of spreading the information from the original data bits throughout a generated codeword, somewhat negating the requirement for an interleaver. Also the properties of an LDPC coded signal can be used to build an estimate of the channel from which the data was received and can therefore be used to equalise the received signal.

2.4 Noise

Noise seen in the HF channel comes from a variety of sources and can be broadly categorised as from Gaussian noise sources, narrowband interferers and impulsive sources [2].

Gaussian noise sources include both man made and natural sources such as *galactic noise*, originating from outside our planet, *thermal noise* generated within all electronics hardware as a function of its temperature and man made noise. In highly populated areas there is obviously generally a higher magnitude of man made noise

than observed in rural or remote areas.

Narrowband interference is generally a result of other radio transmissions being attempted at similar frequencies, the magnitude of narrowband noise observed is dependent on both the geographical location and the time of day; highly populated areas such as central Europe see a congested spectrum as many users attempt are attempting to utilise the HF channel simultaneously [2][11], and usage patterns are liable to change during the course of a day as people go to bed and the range of available HF frequencies changes.

Impulsive sources, which can be modelled as filtered Dirac delta functions, can be either man made or natural. Lightning strikes from nearby storms are a common, natural source of such noise.

2.5 Modelling the HF Channel

Modelling of a communications channel, in the context of a digital, discrete time system, can be accomplished by assuming that the channel transfer function can be instantaneously modelled by a finite impulse response (FIR) filter, $\mathbf{h}[nT]$ (where T is the sampling period and n is an integer) with N taps, where $N = t_c.f_s$, t_c denoting the settling time of the channel when fed an impulse and $f_s = 1/T$ denoting the sampling frequency (or symbol frequency) of the received (or demodulated) signal. The taps of $\mathbf{h}[nT]$ correspond to the impulse response of the channel. A noise function η is then added to the output of $\mathbf{h}[nT]$, and is often assumed to be white Gaussian noise with an amplitude sufficient to model the observed SNR of the signal at the receiver.

Communications channels, such as the HF channel, can be time variant. Long term variations, i.e. those that do not effect any significant changes over the course of a packet transmission, such as changes in the maximum usable frequency (MUF) and lowest usable frequency (LUF), can be assumed to be static when modelling a channel. However short term variations that can appreciably change the parameters of a channel over the course of a packet's transmission time must be accounted for. A common example of such a short term variation is the phase and amplitude variations in the channel resulting from Rayleigh fading and multipath in the received signal. To simulate these short term variations, it is generally required that the taps of $\mathbf{h}[nT]$ change during the course of a transmission. In simple terms, each tap of

$\mathbf{h}[nT]$ can be considered a separate path seen at the receiver (it is possible to have 'padding' taps with a value of zero if the time delay between received multipath components is large compared to its symbol rate), each tap will have a complex value which defines the instantaneous phase and frequency distortion of the received signal component from the corresponding path.

In addition to changing the channel model parameters, Doppler shift of the received signal ought also to be modelled. This is an effect resultant from the time varying altitude of the ionospheric layers and manifests as a frequency shift of the received signal [12]. This can typically be accomplished by multiplying the received signal with a complex frequency.

2.5.1 The Watterson channel model

In his 1970 paper [1] Watterson proposed an ionospheric HF radio channel simulator based on the fact that the properties of a band-limited HF channel change slowly enough that they can be assumed to be invariant over short periods of time (less than ten minutes). The model uses a delay line, tapped at intervals corresponding to multipath delays of a simulated channel. Each tap is multiplied with a complex, normally distributed, random process with a bi-variate Gaussian power spectrum; that is the sum of two Gaussian curves corresponding to the ordinary and extraordinary magneto-ionic components of the path. Figure 2.3 shows the tap gain spectrum of a Watterson model tap gain function, where σ_{si} and ν_{si} denote the standard deviation and centre, respectively, of each Gaussian function. The random processes can be modelled by filtering white Gaussian noise through a Gaussian filter. Suitable tap values are found by taking the inverse Fourier transform of the Gaussian function. When these taps are convolved with a signal in the time domain, a Gaussian Doppler spread is applied to the signal. See equation 2.1 [13] where the desired Doppler spread d_j is twice the standard deviation.

Figure 2.4 shows a block diagram of the Watterson model. The incoming signal is fed into a tapped delay line, each tap is multiplied by an independent complex tap gain function, and the result is added together giving a multipath Rayleigh faded output. It is necessary to input an analytical (complex) signal into the system, which for real signals requires a Hilbert transform filter placed in series before the input to generate the imaginary part.

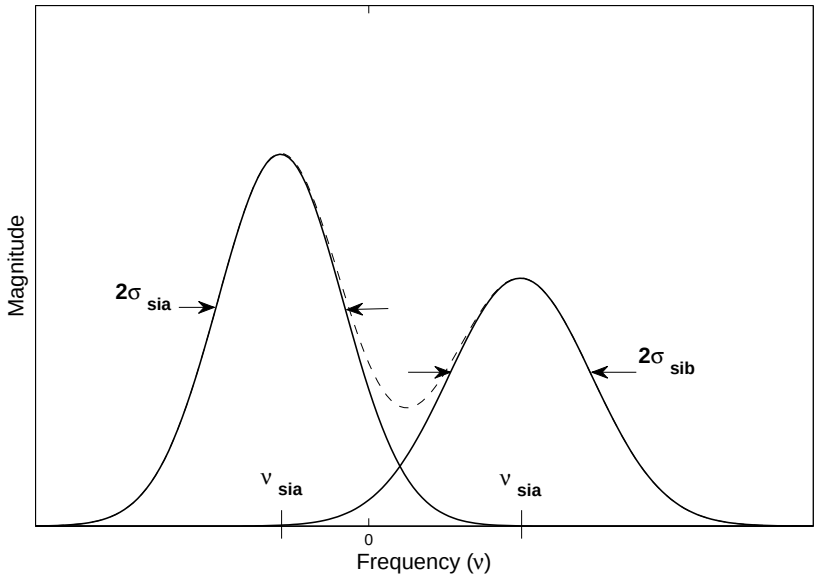


Figure 2.3: Spectrum of Watterson tap gain functions [1]

$$f_j(t) = \sqrt{2}e^{(\pi t d_j)^2}, -\infty < t < \infty \tag{2.1}$$

This equation represents a Doppler spectrum for each tap, where the signal's spectrum is 'smeared' over a range given by the width of the Doppler spreading function. As it is randomly time variant, complex multiplying (mixing) it with an incoming signal results in Rayleigh fading [10]. This results in amplitude fades, phase disturbance and inter-symbol interference in the affected signal. It is a common model for narrowband communication signals without a dominant line of sight component. In the context of HF, it adequately models the atmospheric scattering of a signal on a single path over long distances.

2.6 ITU HF Channel Recommendations

The CCIR (Comitee consultatif international pour la radio, later inaugurated into the international telecommunications union) recommended parameters for a Watterson channel simulator to simulate good, moderate and poor quality HF channels (ITU-R recommendation F.520-2 [4]). These are defined as multipath channels with two paths separated by a given differential time, the parameters of which are given in table 2.1. Both paths have equal mean gains and equal frequency spreads. Additive

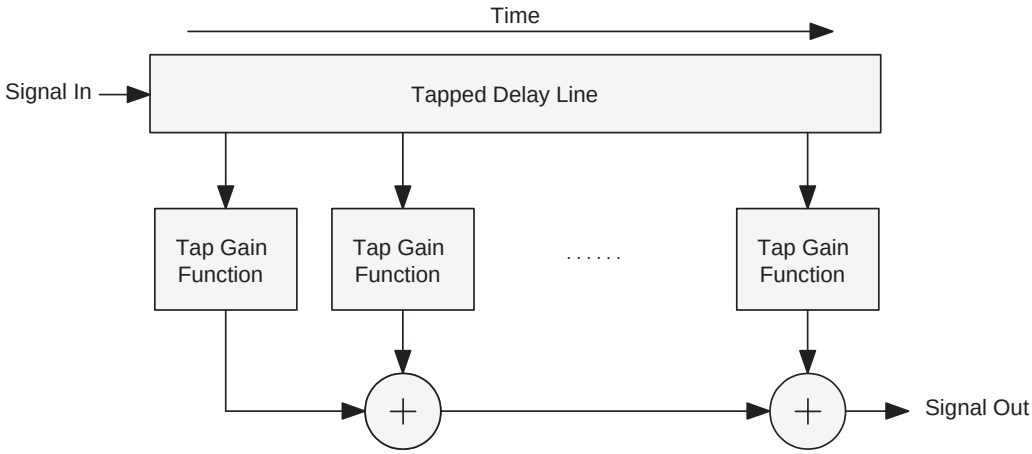


Figure 2.4: Block diagram of the Watterson model

white Gaussian noise (AWGN) is added before the output of each path. The bit error rate of a tested communication system is taken against the ratio of the energy-per-bit to the noise density (E_b/N_o). These parameters provide measures for the qualitative analysis of HF communication systems and are popular enough to allow for quantitative comparisons against other techniques. Figure 2.5 shows the gain characteristics with respect to time of the three recommended channels, the good channel exhibits the longest periods of time between deep fades, in the order of tens of seconds, the poor channel in comparison fades quickly and often, with multiple deep fades a second.

Parameter	Good	Moderate	Poor
Doppler Spread (Hz)	0.1	0.5	1.0
Path Delay (ms)	0.5	1.0	2.0

Table 2.1: The channel model parameters from ITU recommendation F.520-2 [4]

A later ITU recommendation, F.1478 [14], has superseded the F.520 recommendation, adding parameters for high, low and mid latitudes each in quiet, moderate and disturbed conditions. Equivalences to the F.520 document are, however, noted in the document and therefore the F.520 defined channels are still quite adequate for most applications.

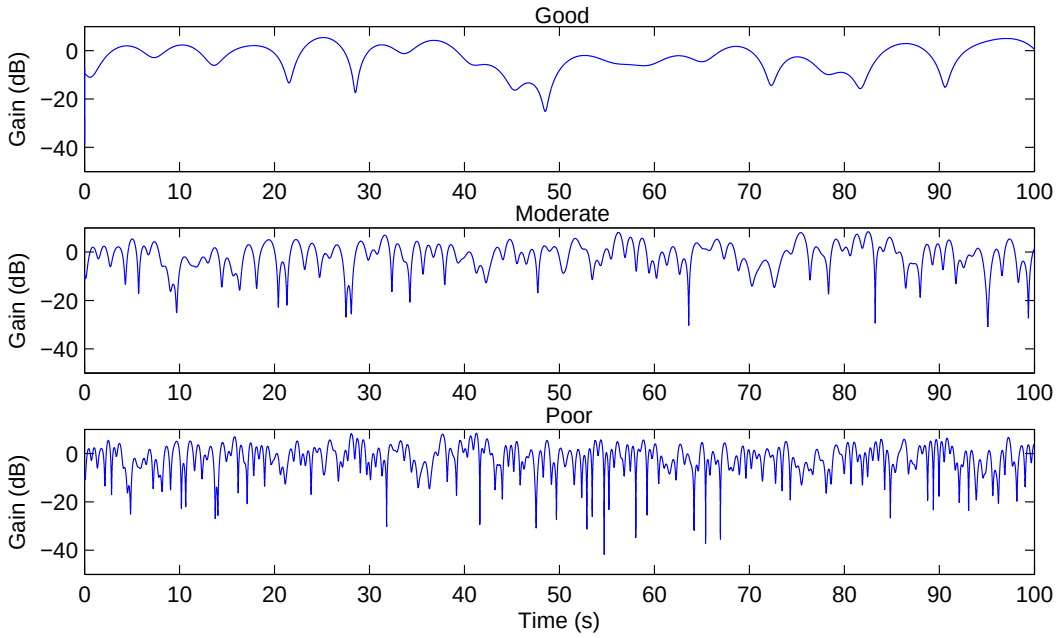


Figure 2.5: Gain over time characteristics of ITU recommended HF channels

2.7 Development of an RF frequency HF channel simulator

A Watterson model was implemented using GNU Radio and a Universal Software Radio Peripheral (USRP) [15] to create an affordable, versatile channel simulator capable of real time channel simulation at HF frequencies. The system's principle of operation is to use the USRP to receive an incoming HF signal and convert it to baseband, apply the channel function at baseband and mix back up to HF frequency for transmission at the output. The USRP architecture allows for full-duplex communications and signal processing is performed in real time on a PC.

Figure 2.6 shows a functional block diagram of the GNU Radio implementation of a Watterson model. It is functionally equivalent to the block diagram given in Figure 2.4 with two paths. The implementation of the complex taps is also shown, which is described in the next section.

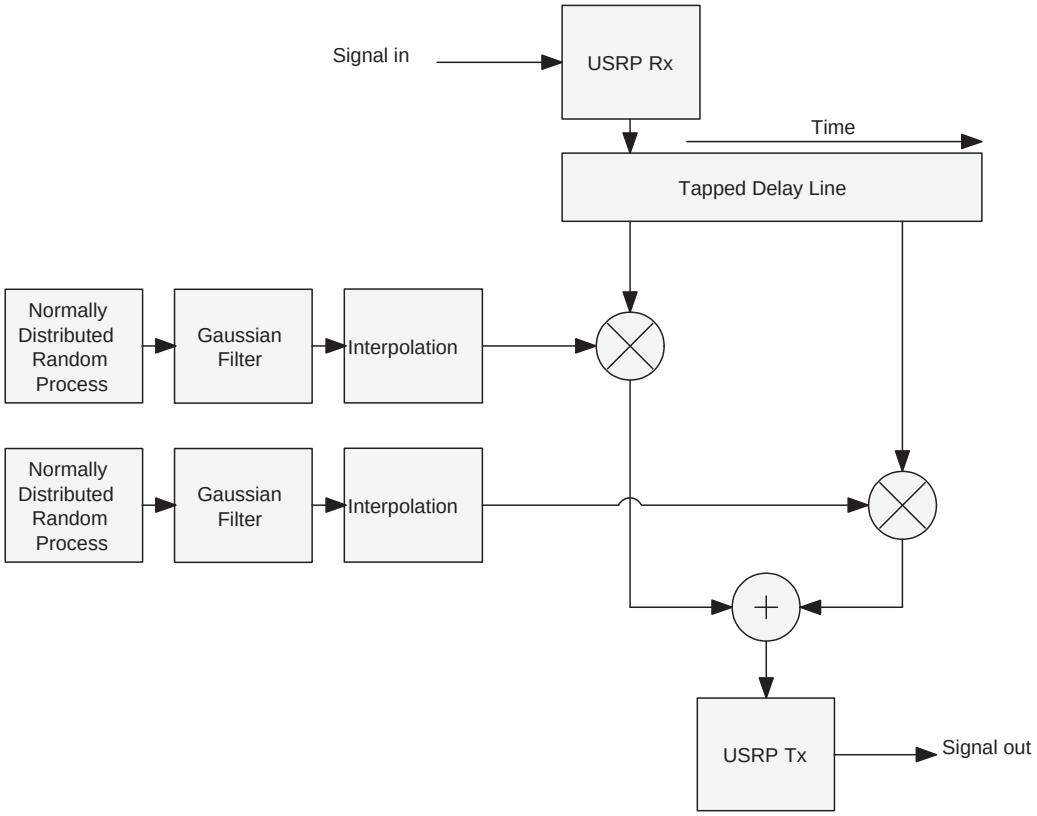


Figure 2.6: Block diagram of the GNU Radio Watterson model implementation

2.7.1 Complex Taps

Equation 2.1 gives values for the taps of a Gaussian filter such that the elements h_n of the vector of taps **h** are given by the equation

$$h_n(t) = ke^{-(\pi f_j n T_s)^2}, -N < n < N \tag{2.2}$$

where T_s is the sample time of the system, the constant k maintains a power gain of 1 through the filter, and N denotes the length of the filter, where N must be sufficient to allow the tails of the Gaussian function to reduce to an arbitrarily small value, recommended as at least 0.01 of the value of the largest tap [13]. For a given Doppler spread of f_j the required value of N therefore increases proportionally with the sampling frequency f_s. To avoid excessively long filter lengths and their associated processing power overheads a lower sampling frequency f_d is used, where $f_s = Df_d$, with the resultant Doppler spectrum up-sampled to the baseband sampling frequency. The decimation parameter D must be chosen low enough to fit the bandwidth of the filtered signal within its Nyquist limit.

The constant k is used to maintain a noise power gain through the tap function of

1; to find this value it is necessary to find the power, P_n of the noise signal out of the Gaussian filter, which is the power of the white noise signal multiplied by the power gain of the filter. The transfer function of the Gaussian filter, equal to the taps function, generated by GNU Radio is

$$H(f) = \frac{f_d\sqrt{\pi}}{f_s} e^{-(\pi f_d f)^2} \tag{2.3}$$

and its power gain is found by integrating its square between -infinity and infinity.

$$\begin{aligned} G_n &= \int_{\infty}^{\infty} |H(f)|^2 \, df \\ &= \frac{f_d\sqrt{\frac{\pi}{2}}}{f_s^2} \end{aligned} \tag{2.4}$$

The white Gaussian noise generated by GNU Radio is of the form $n(t) = n_r(t) + n_i(t)j$, where $n_r(t)$ and $n_i(t)$ are normally distributed random numbers with a variance of one. Therefore it has a constant power spectrum of

$$P_n = 2f_s \, W.Hz^{-1}. \tag{2.5}$$

The power of the signal at the output of the Gaussian filter is therefore

$$P_n G_n = \frac{2f_d\sqrt{\frac{\pi}{2}}}{f_s}. \tag{2.6}$$

To maintain a constant power gain of one through the tap gain function, the following gain must be placed at the output of the Gaussian filter

$$k = \sqrt{\frac{1}{P_n G_n}} = \sqrt{\frac{f_s}{2f_d\sqrt{\frac{\pi}{2}}}}. \tag{2.7}$$

2.8 Verification of the GNU Radio channel simulator

The channel simulator was designed in accordance with Furman's recommendation for an improved HF simulator specification [13] and all requirements are met or exceeded.

2.8.1 Output statistics

The output of the simulator, when fed with a single tone input, with one path and no AWGN, should be *Rayleigh distributed* which means its probability density function should be given by

$$p_r(y) = \frac{2}{P_s} exp\left(-\frac{y^2}{P_s}\right) \tag{2.8}$$

where P_s is the power of the scattered signal.

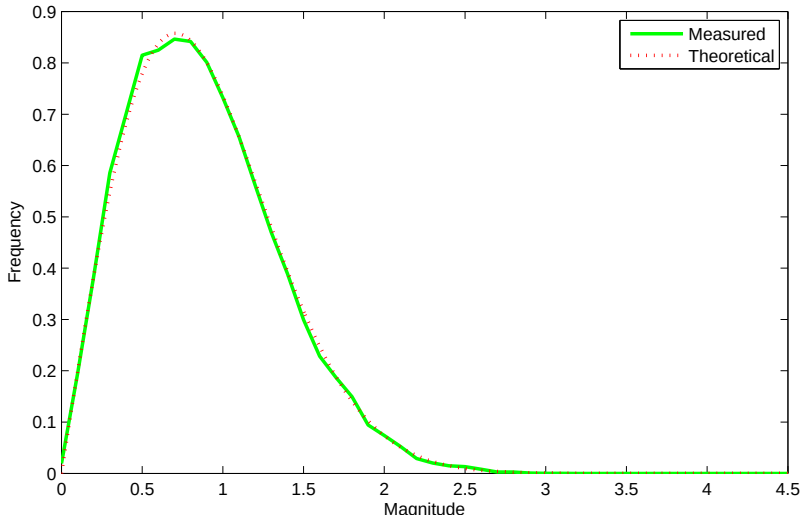


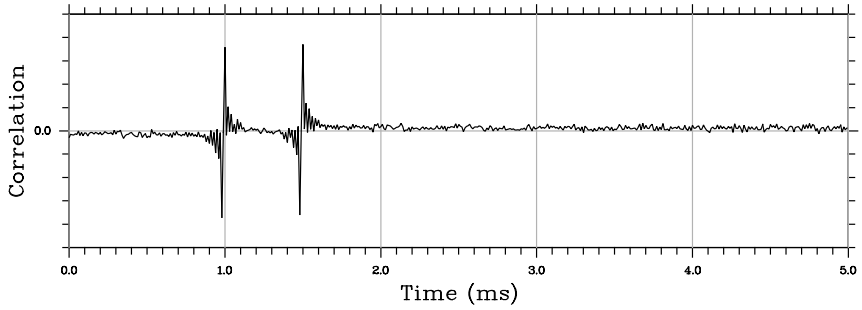
Figure 2.7: Single path output distribution

To perform the test the simulator was set up to execute with a long vector of ones as the baseband input and a single channel with 50 Hz of Doppler spread, output was recorded as a .wav file and subsequently imported and analysed in Matlab. Figure 2.7 shows the probability density function of the sampled output with the ideal theoretical response as given in equation 2.8.

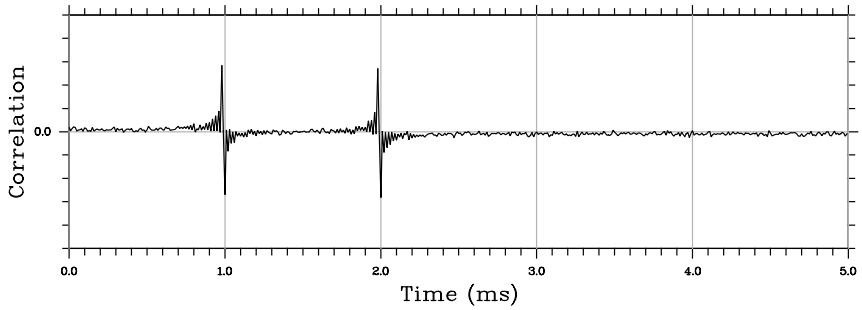
2.8.2 Verification of Discrete Delay

When the channel simulator is simulating a multipath channel its output is the sum of two independently Doppler spread and time delayed versions of the input. It is important that this metric be accurately controllable to simulate a wide variety of channel conditions. The path delay functionality is controlled by a tapped delay line, which was implemented as a signal processing block in the simulator written in the C programming language.

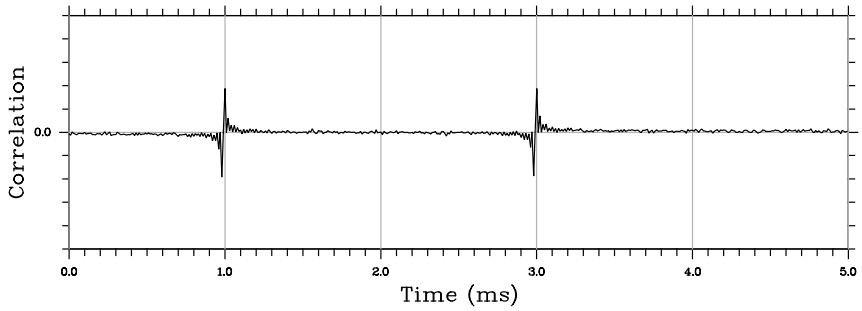
The multipath delay times were verified by sending a known wide spectrum signal through the simulator and then correlating it with the output of the simulator. A random noise source was chosen as the wide band test signal. Figure 2.8 shows the results obtained from the test for the good, moderate and poor CCIR channels. The result shows obvious spikes at the correct path delays, showing that the delay line in the simulator is working as expected.



(a) CCIR Good



(b) CCIR Moderate



(c) CCIR Poor

Figure 2.8: Cross correlation of input and output signals

2.8.3 Verification of Frequency Spread

The Doppler spread is controlled by the tap gain functions which multiply the signals from the tapped delay line by a Gaussian low pass filtered normally distributed random process. To test their functionality a sine wave was fed into the USRP simulator, and the result was captured on a spectrum analyser. Figure 2.9 shows the output of the simulator with spreads of 25 Hz, 50 Hz and 100 Hz. Figure 2.9(a) shows the results of a sine wave control test. This is essentially the result of programming GNU Radio to feed the input of the USRP straight back into its output. It should be observed also that these simulations are conducted in the HF frequency band (at RF), demonstrating that the simulator is capable of simulating at these frequencies. The tests were all performed using two simulated paths.

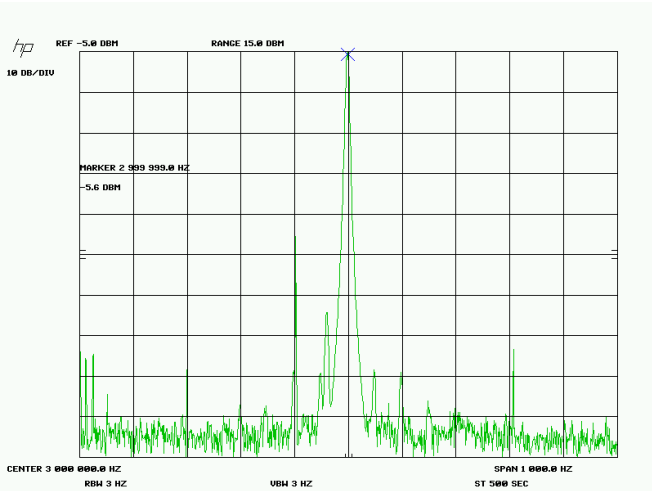
The results show that the frequency spread from the simulator is accurate, it is clear from Figure 2.9 that the 3 dB bandwidth of each simulated channel is equal to the programmed spreading frequency. The frequency distribution is also correct, following a Gaussian distribution. There are some spurious peaks visible on Figures 2.9(a) and 2.9(b), the source of these was not discovered.

2.9 Further Work

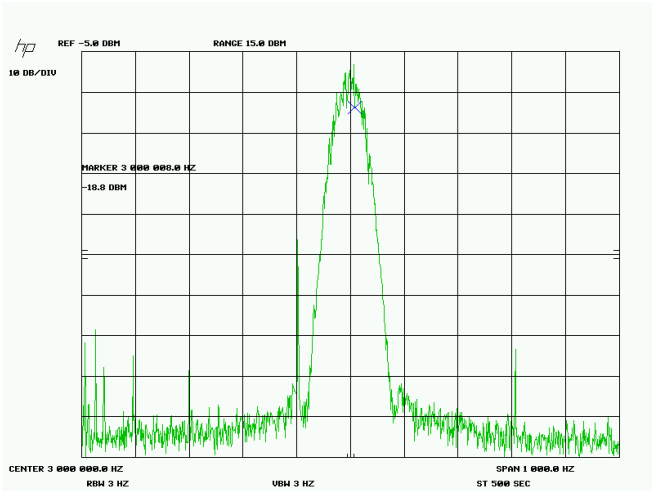
The channel simulator at present has been tested with a maximum of two paths but this number can theoretically be increased indefinitely (computer processing resources allowing). The Doppler spread was correct for the values tested but further verification of the simulator could take place at very low spreading frequencies ($<$1 Hz). This will require the use of a very high resolution spectrum analyser. The spreading function itself can be examined and the Gaussian low pass filter improved to give bi-variate Gaussian output (as described by Watterson [1] and seen in Figure 2.3).

The GNU Radio platform itself should be able to support more complicated channel models that may be more suitable for wider band communications system. At wider frequencies the Watterson channel model becomes less representative of a real HF channel, due to it not modelling frequency selective fading, and therefore could be replaced in preference of a more appropriate model. The processing power, adaptability and useable RF bandwidth of a GNU Radio platform makes it a compelling

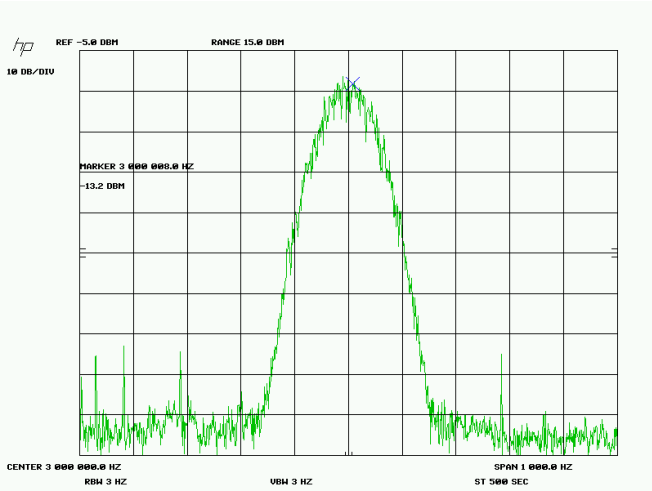
choice for the implementation of such wide band channel models.



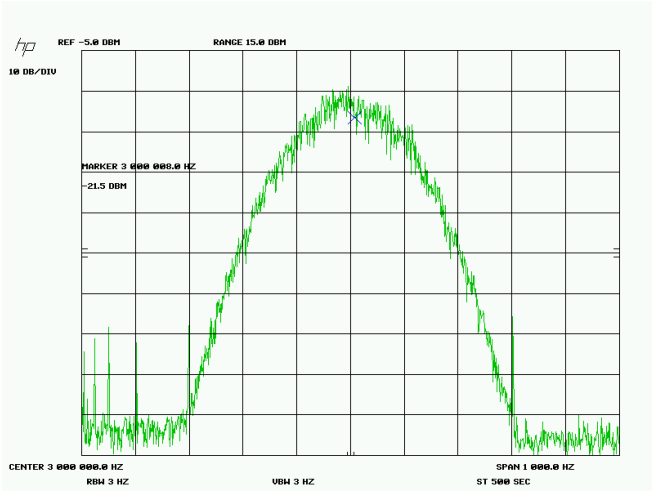
(a) Sine Wave Control



(b) 25 Hz spread



(c) 50 Hz spread



(d) 100 Hz spread

Figure 2.9: Spectrum Analyser Plots of Simulated Doppler Spreads

Chapter 3

HF Standards

This chapter describes the common standards used by amateur and military communications systems. There are a multitude of standards that have been defined, which typically have a low data rate and high robustness to poor channel conditions and noise compared to other wireless standards (such as IEEE 802.x). This is due to the low available bandwidth and availability in the HF band, and the difficult channel conditions encountered.

3.1 Historical and amateur standards

Some of the first HF communications systems were radio teletype (RTTY) type devices, which uses a binary data modulation technique, encoding characters as 5-bit codes and transmitted data on two tones, one for a '1' and another for a '0', effectively a simple frequency shift keyed (FSK) system. The tones are typically spaced 170 Hz apart. The baud rate is typically 45.45 symbols/s, allowing approximately one word a second, the spectral efficiency of this technique is, however, very poor for its data rate. The frequency tones are spaces 170 Hz apart, resulting in a signal bandwidth of approximately 200 Hz. The 5-bit codes allow only a limited number of characters to be transmitted, but a 'shift state' is used to double the number of available characters by sending a special code. There is no error correction used as standard with this technique.

Hellschreiber is a technique introduced in the 1920s [16]. Known as facsimile-RTTY (FAX-RTTY), it encodes data into pixels, in a similar way to how a fax machine works.

It is capable of sending text and pictures and originally used the standard RTTY waveform for sending pixel values. In the 1990s Hellschreiber began to be used over different modulation formats such as phase shift keying (PSK) and multiple frequency shift keying (MFSK).

Multiple frequency shift keying (MFSK) [17] was introduced in the 1960s, originally as 'Piccolo' by the UK government. It is a variation of FSK which uses multiple frequencies to represent the bits of a data transmission. The tones are used to generate orthogonal symbols, such that for M frequency tones each symbol can typically carry $\log_2 M$ bits of data. For this reason M is normally chosen to be the square of an integer. MFSK8 and MFSK16 (which use 8 and 16 tones respectively) were introduced for use by the amateur radio community. Later, 'Olivia' MFSK was introduced which was optimised for use in very poor channel conditions, at SNRs as low as -10dB. Olivia is currently one of the more popular standards used by radio amateurs in very poor SNR conditions.

Simplex teletype over radio (SITOR, referred to by the amateur radio community as amateur teleprinting over radio, or AMTOR) [18], is a technique introduced in the 1980s, but has been out of use since the 1990s, having been superseded by packet teletype over radio (PACTOR) [16]. It uses 7-bit codes which are sent at 100 baud, using the same 170 Hz spacing as RTTY, and uses error correction on the transmissions.

PACTOR was developed from SITOR and packet radio, which packetises transmitted data. It uses the same FSK modulation as RTTY, but the packetisation process provides increased bandwidth efficiency. It also uses the error correction techniques used with SITOR/AMTOR. When PACTOR was released in 1991 it quickly superseded SITOR/AMTOR. The PACTOR II and PACTOR III [16] variants are faster, enhanced modes which utilise data compression. They are licensed by SCS, the German company which initially introduced the standard.

Clover and G-TOR are two standards that were introduced to compete with PACTOR [16]. They use phase shift keyed modulation and are both faster than PACTOR, with better error correction, allowing them to perform well even under poor HF conditions.

PSK31 is currently one of the most popular standards used by the HF amateur radio community [16][19]. It uses binary phase shift keying (BPSK) at a data rate of 31 baud, hence the name. Its popularity stems from its very narrow bandwidth com-

pared to most other standards, which allows many users to use the limited available HF bandwidth and allows very low power operation. It has, however, a relatively slow data rate. There is no error correction as standard on PSK31. There are a number of other PSK standards, all denoted by PSKN, where N denotes the baud rate. These include PSK5, PSK10, PSK63 and PSK125. There are variants of these standards with 1/2 rate forward error correction added, which are denoted by adding an 'F' onto the end of the name. PSK63F, for example, uses error correction and has approximately the same data rate as PSK31.

Multi-tone 63 (MT63) [16] is an amateur radio standard which uses orthogonal frequency division multiplexing (OFDM) to modulate data onto 64 parallel tones. Forward error correction is added to the data as standard. It is very fast compared to the other HF standards, and resistant to poor channel conditions. However it also uses a much wider bandwidth, and can drown out other transmissions. For this reason it is not commonly used by radio amateurs, and is more suited to live operations that require a high data bandwidth.

3.2 ITU-R radio recommendations

The International Telecommunications Union (ITU) is an international body which is responsible for standardising, disseminating and overseeing the development of telecommunications technology worldwide. The radiocommunications sector is known as the ITU-R, and originates from the *Comit Consultatif International pour la Radio* (Consultative Committee on International Radio or CCIR), which became the ITU-R in 1992. The ITU-R publish recommendations for the design and simulation of radio communications systems, including those for use in the HF band.

3.2.1 Ionospheric channel simulation recommendations

The ITU-R (formerly CCIR) recommendation F.520-2 [4] is a well cited document detailing recommendations for the design and implementation of a HF channel model to simulate narrow-band ionospheric channel conditions. Watterson channel model [1] parameters such as duration, depth and frequency of fades, suggested Doppler shifts and multipath delay times of a channel simulator are suggested, and

a number of channel models are defined, including the ITU-R ‘good’, ‘moderate’ and ‘poor’ channels, which are widely cited and used to benchmark the throughput of modulation, equalisation and error control techniques in the literature. The existence and wide use of these channel models provides a good basis for deriving performance comparisons between such techniques, and also provide a reasonable estimate of how they might be expected to perform in various real channels.

In 2000 ITU-R recommendation F.520-2 was officially withdrawn and replaced by recommendation F.1487 [14] which expands upon the former publication. In this document values are defined for channel models representative of disturbed, moderate and good conditions in each of low, medium and high latitudes. Three of these models are equivalent to the original ‘good’, ‘moderate’ and ‘poor’ channels defined in recommendation F.520-2. This document additionally provides expected performance surfaces of simple modems using the described simulators.

Another interesting recommendation, useful for the design of simulated channel models, is P.372-8 [2], which describes and characterises radio frequency noise in links, and provides average measured values. Many sources of noise are considered including man-made interference, atmospheric and galactic noise, and comprehensive descriptions of and models for these noise sources are postulated. Noise Figures are given as functions of frequency and for different geographic locations.

3.3 MIL-STD-188-110

The US military standards section 188 (MIL-STD-188) are interoperability standards defined by the US military for compatibility between radio communications devices. Of particular interest to the this project is the standard for interoperability between data modems, MIL-STD-188-110B [20]. This is a very widely used standard for digital communications systems and is widely cited in the literature for performance testing of new techniques.

MIL-STD-188-110B defines a number of waveforms including defining packet formats, modulation techniques, interleaver lengths and other parameters of interest. Waveforms are defined which support bit-rates from 75-12800 bps. Single-tone modulation techniques chosen include frequency shift keying (FSK), phase shift keying (PSK), quadrature amplitude modulation (QAM). A 39-tone modulation format is

also proposed for using parallel tone techniques. The standard has been designed for a 3 kHz (voice channel) bandwidth.

Forward error correction coding is defined for use with each bit rate, this ranges from a code rate of 1/16 for the lowest bit rates to no coding on the transmitted data. The suggested codes to be used in the standard are tail-biting punctured convolutional codes, although other codes can be used.

MIL-STD-188-110C is an updated standard, and at the time of writing it is still in the process of being developed. It is expected that this standard will build upon the current (110B) standard by expanding maximum channel bandwidths up to 24 kHz, and correspondingly increasing supported bit rates.

3.3.1 Packet format

The following describes the packet format used for the serial tone waveforms of MIL-STD-188-110B as it relates directly to some of the work performed during this project. Figure 3.1 shows the MIL-STD-188-110B single-tone packet structure. At

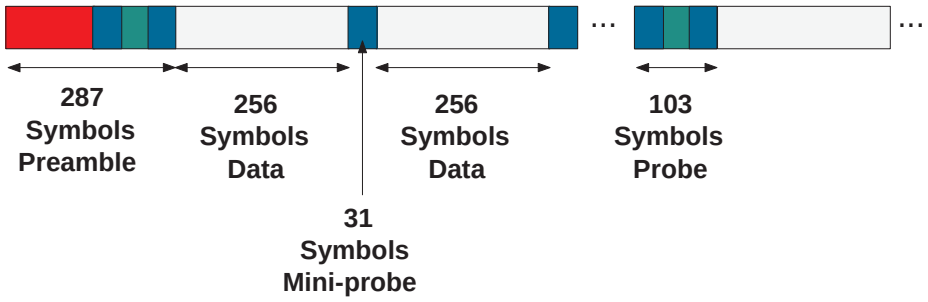


Figure 3.1: The MIL-STD-188-110B single-tone packet structure

the start of a transmission, a 287 symbol preamble is transmitted, then transmission of 'frames' commences. One frame consists of 256 data symbols and a 'mini-probe' of 31 symbols. After 72 frames have been transmitted, a 72 symbol subset of the initial preamble is inserted into the transmission. The combination of the mini-probe at the end of the 72nd frame and these extra 72 symbols forms a 103 symbol probe sequence. This probe is inserted every 72 frames. The preamble, probe and mini-probe features of the packet are used by the receiver to detect, synchronise and equalise the packet and thus maximise the probability of successful receipt of the data.

3.4 NATO HF STANAGs

The North Atlantic Treaty Organisation (NATO) maintains and releases a number of documents that define common properties and parameters that various military devices require. They are termed ‘standard agreements’ or STANAGs for short. For radio communications devices they typically define standards for interoperability of devices such as modulation techniques and waveform definitions. Some of the relevant STANAGS are detailed below.

STANAG 4197 [21] defines the required characteristics for a modem used to transmit digital voice data at 2.4kbps. It defines a 39-tone parallel tone waveform be used for data with a subset of 16 tones used to transmit a 240-bit transmission synchronisation preamble. Each tone transmits two bits per frame (QPSK encoding). A coding specification is defined based on Golay codes. The document concludes with an evaluation of the performance of the modem in noise and with varying Doppler shift.

STANAG 4205 [22] defines technical standards for hardware used in military HF communications devices. It covers minimum acceptable performances of various subsystems such as the stability and phase noise of local oscillators and the linearity of amplifiers. This standard could form a good design basis for any HF hardware.

STANAG 4285 [23] defines characteristics for serial tone modems operating at 1200, 2400 and 3600 bps. The waveforms are all identical save for the type of modulation that is used (binary-PSK, quadrature-PSK and eight level-PSK respectively). A frame structure is defined for a packet length of 256 symbols, where 128 are data symbols, 80 are synchronisation symbols and 40 are reference symbols (equivalent to the mini-probe symbols observed in the MIL-STD-188-110B waveform). Figure 3.2 shows the details of the packet structure. It is evident with this packet that the overhead for transmitting the preamble is 50%, which is higher than that of the MIL-STD-188-110B packet structure.

STANAG 4285 continues to perform a thorough evaluation of the error performance of an example modem in AWGN and using the ITU-R moderate and poor channels without any forward error correction coding. Further to this, the relationship between the error rate of the waveform and the Doppler spread of the channels is also graphed.

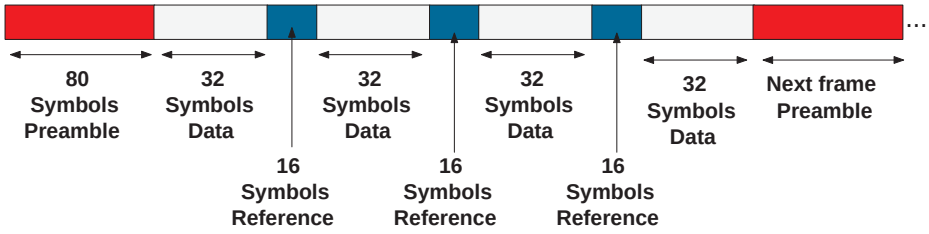


Figure 3.2: The STANAG 4285 single-tone packet structure

STANAG 4415 [24] defines waveform standards for HF modems operating in very poor channel conditions. It is designed for 75 bps data transmissions and is, in fact, identical to the 75 bps mode of the MIL-STD-188-110A specification. The waveform is specified to give low bit error rates at low SNRs of approximately 0dB even with very high values of Doppler spread within the channel (40-50 Hz). Helpfully this document provides an appendix with the code required to implement this waveform in the C programming language.

STANAG 4538 [25] defines characteristics of a standard automatic link establishment (ALE) system for use to ensure interoperability of military HF communications devices. Its scope is set at a somewhat higher level than the remainder of this project, which primarily investigates lower level modem functionality such as error control coding and equalisation.

STANAG 4539 [26] defines general non-hopping HF waveforms for links from 75 to 12,800 bps. The waveforms inside are designed to be interoperable with the MIL-STD-188-110B standard described above, and the two documents are very similar. The STANAG 4539 serial waveform is identical to the MIL-STD-188-110B waveform given in Figure 3.1. The document gives a performance evaluation of the different combinations of parameters possible for a modem system, these data are very similar to those presented in [20].

3.5 Digital Radio Mondiale

Digital radio Mondiale (DRM) [6] is a recent digital radio standard that provides standards for digital radio transmissions up to 174 MHz, using typical channel bandwidths of 9 to 10 kHz, but with support for narrower bandwidths of 4.5 to 5 kHz and wider bandwidths of 18 to 20 kHz. It is positioned to overtake analogue AM and FM radio signals by providing much better sound quality for the same bandwidth.

DRM uses coded orthogonal frequency division multiplexing (C-OFDM) [27] which is a modern, robust modulation technique that provides a good spectral efficiency and resistance to noise and channel conditions. The OFDM technique is discussed in section 6.3. The coding technique used is multilevel coded modulation (MLCM), which is similar to trellis coding and effectively maps data to symbols by means of a trellis, whereby each symbol is determined by the current data bit and a number of previous symbols.

DRM specifies four different modes for different channel conditions. Mode A, specifying 19.6 to 30.8 kbps in 9kHz, is for high speed transfer with benign channel conditions and line of sight links. Mode B, specifying 11.6 to 14.5 kbps for 16-level quadrature amplitude modulation (QAM) modulation and 17.4 to 27.4 kbps for 64-QAM modulation in 10 kHz, is for longer range single hop paths typical of international European links. Mode C, specifying 9.1 to 11.4 kbps for 16-QAM and 13.7 to 21.5 kbps for 64-QAM in 10 kHz, is for multi-hop transcontinental links. The most robust is mode D, which specifies 6.9 to 7.5 kbps for 16-QAM and 9.1 to 14.3 kbps for 64-QAM in 10 kHz, is suitable for very long multi-hop paths and near vertical incidence skywave (NVIS) use. Modes A and B are suitable for use at all frequencies and can carry high definition audio, while modes C and D are used solely at short-wave frequencies and with their higher level of protection, lack the capacity to carry audio of sufficient quality to rival AM broadcasts and are more suitable for lower speed data links.

The DRM specification additionally specifies source coding techniques for encoding audio streams such as advanced audio coding (AAC) and MPEG (motion picture experts group - responsible for a number of well-known video standards), which provide different levels of audio service. Multiplexing is also specified, such that a transmission can embed extra information such as signal descriptors and text messages.

Chapter 4

System Requirements

This chapter defines the system requirements for the remote and base station units that would be used in the final system. These are largely based on some roughly defined specifications from the sponsor, inferences based on these requirements, and the assumed application of the project results.

4.1 Project Aim

The main aim of this project was to conduct an investigation into modern digital HF radio communication techniques with a view to identifying suitable, modern techniques and designing a radio communications system suitable for transmitting data from a battery powered device at a remote location to a mains powered device resident in a base station without reliance on existing infrastructure. The system is proposed to work for over-the-horizon links at ranges of hundreds of kilometres.

It was desired that a broad range of modern communication techniques be examined and evaluated, and that techniques suitable for the purposes of the specified link be identified, performance tested, and integrated into a communications system. The nature of the project has meant that a wide range of possible research directions were available, and time constraints necessitated that the project focus on a few, promising techniques that are pertinent to the particular requirements of this project; namely power efficiency of the remote unit, system performance in low signal-to-noise ratio (SNR) conditions, and the asymmetry of the link in terms of data throughput. To this end the project has focussed on error control techniques

and compensation for the effects of the HF channel on the signal, and how they can be used to maximise link performance under the particular specifications of the modems.

It was decided that a demonstration unit be designed and manufactured to show the techniques discussed in the project working in the real world. By moving from simulations into the real world problems that may not otherwise be considered are found, and real links can be used to evaluate the system performance. The test unit was designed and manufactured to conform to the specifications given below.

4.2 Remote unit requirements

The remote unit is the battery powered hardware which will be deployed at a remote location. Its main purpose is to transmit data to the base station. A low data-rate receive chain will be required, which will be used for receiving instructions and data repeat requests sent from the base station to the remote unit. Following is a list of requirements described by the sponsor in a meeting held in the first year of the project [28]. The specifications were kept deliberately vague so few restrictions were imposed on the direction of research.

- **Size:** Physically Small.
- **Power Source:** Low power operation from 4 alkaline D cells.
- **Operating schedule:** Opportunistic, with a minimum operating period of 1 week, with 1 month being highly desirable.
- **Antenna:** Operating with a poorly matched, low gain antenna (short or even buried wire).
- **HF waveform:** Designed for low probability of detection.
- **Data:** Five A4 pages of text per day.

The following subsections analyse these requirements and attempt to quantify them, wherever possible, to metrics to be used in the research and design of the system.

Figure 4.1 shows a block diagram of the remote unit showing the functional blocks that were designed during the course of this project. It shows the basic transmit

chain, which includes forward error correction (FEC) and automatic repeat request (ARQ) subsystems. Suitable algorithms for these blocks are discussed in chapters 7 and 8

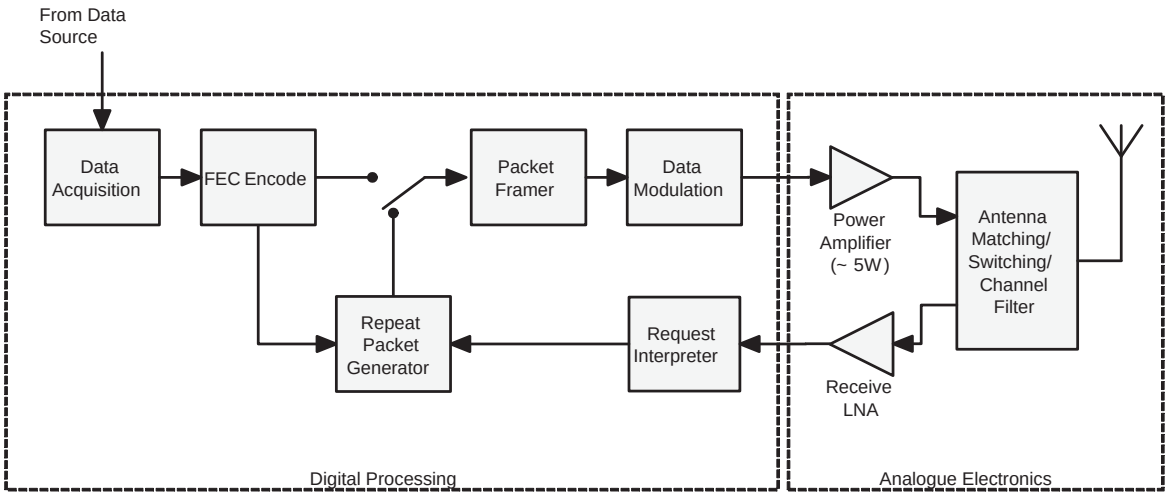


Figure 4.1: Remote unit functional block diagram.

4.2.1 Size

The field unit is specified to be 'physically small', from which it can be inferred that the unit ought to be easily portable, with room to accommodate the electronics and the battery power supply. The four alkaline D cells specified for the power source gives an idea that this piece of equipment ought to be about the same size and weight as a standard battery powered desktop radio that could be purchased from a high street retailer. To tentatively quantify this requirement; dimensions of approximately 20x20x10cm ought to be the maximum allowable size.

4.2.2 Power source and data rate

The power source is specified as 4 D-type alkaline cells. This configuration is capable of providing 15 Ah at 6 V [29], which provides metrics for the expected energy and voltage available at the power source. While alkaline batteries are specified, in the finished system it may be desirable to use lithium-ion cells instead, for their higher energy density (Li-ion cells have been announced with energy densities approaching 675 Wh/l [30] compared to alkaline D-cells at 322 Wh/l and AA-cells at

420 Wh/l [29]).

The data rate of the system is given as five A4 sheets of text per day. The capacity of one A4 sheet in this report is approximately 3000 characters including spaces. Assuming 8 bits per character the total data to be sent is approximately 15kB per day. This corresponds to approximately 100 kB of information transferred per week, or 400kB for a month of operation.

As the requirements define both the energy available to the remote unit from the batteries and the total required data to be transmitted after deployment (five A4 pages a day for one week minimum for up to a month if possible), a target data rate can be inferred if a metric is available to quantify the energy required per transmitted bit. It is not an unreasonable assumption that the transmit power amplifier (PA) will draw in the region of 5 W of power during operation as this Figure is commonly used by the QRP community for HF transmissions [31]. Assuming the rest of the circuitry will consume 2 W, the total power draw of the system will be in the region of 7 W. The energy contained in the specified batteries is given in [29] as 15 Ah at 6 V, or 90 Wh, this corresponds to approximately 13 hours of continuous transmission. To transmit 400 kB in this time with a 1/2 rate error correction code would require a continuous data rate of less than 100 bps. This estimate neglects the power required by the modem when it isn't transmitting (i.e. when it is sleeping or listening), and the margin required to retransmit data in the event of an error in its receipt at the base station. Additionally, the 5 W transmit power Figure might prove to be too low when attached to a poor antenna. While the theoretical required data rate specification is low enough to seem easily achievable, only testing of a real-world system will prove the feasibility of the project. A provisional target data rate of 1 kbps was kept in mind during the design of the system.

4.2.3 Receive chain requirements

There is a requirement to implement a feedback communications path from the base station back to the remote unit. This link will not be required to transmit any data, only instructions from the base station, for example to initiate a transmission, or repeat requests during communications. As such the receive chain could be con-Figured to receive a large number of different signals. Chirps or pseudo-random signals could be used to encode instructions from the base station. This opens up the possibility that the transmitter could derive a channel estimate during receipt of

these commands and adapt its transmission characteristics to suit. The channel characteristics are likely be similar in either direction (as the path the signal takes is likely to be geographically similar). For example, if the received SNR is detected below a certain threshold the rate of any FEC coding can be increased to provide further redundancy to protect against data errors. This idea was not explored further during the project.

There has been little work performed in designing the feedback link, the techniques discussed in later chapters that require it (such as ARQ) have been based on the assumption that a simple feedback link will be available. The remote unit hardware as presented in chapter 10 includes a receive chain design and discussion.

4.2.4 Antenna Requirements

The antenna has been specified to be have a low gain and poor electrical match. Given the nature of the project and speculating on the environment in which it is to be deployed, the antenna can be assumed to be opportunistically placed and have a low visual impact (it would be undesirable to erect a large antenna in a national park, for example). This might mean the antenna may be deployed in a tree or even underground. A requirement of the system, and in particular the transmitter electronics, including the power amplifier, must be developed to account for a possibly very poor match at the antenna. A poor match will result in signal energy being reflected back into the input of the amplifier, which has implications for the design of the power amplifier.

Following from the assumed very poor quality antenna and transmit power limitations, the signal strength at the transmitter can be assumed to be poor and therefore the signal to noise ratio at the receiver will be low. This impacts the signal detection, equalisation and error correction techniques chosen for use at the receiver, which must be optimised for low SNR conditions.

To gain an idea of the properties of a poor antenna a simple underground antenna assembly was constructed and characterised in a variety of ground conditions, the results of these tests and a discussion are presented in chapter 9.

4.2.5 Processing requirements

The remote unit requires a processing unit to perform baseband processing of the received and transmitted signals. The processor is required to have the computational capacity to perform the functions required to generate the output data signal, including forward error correction (FEC) encoding, repeat packet encoding, data modulation, re-sampling and filtering of the output signal. It should also have sufficient capacity to receive commands sent from the base station (which imposes requirements on the techniques used to encode these commands). The low power requirement of the board entails that the algorithms used in the remote unit for performing the above operations be a computationally simple as possible. By keeping the required processing capacity of the remote transmitter to a minimum, the lowest possible processor clock speed can be used at the remote unit which minimises power consumption. Throughout the project algorithms have been chosen so as to minimise the demands put on the transmitter without compromising on system performance.

4.3 Base station requirements

Similarly to section 4.2, some provisional requirements for the base station were also specified during the meeting with the sponsor [28]. These are listed below.

- **Size:** Not an issue, can be housed in a 19” rack if necessary.
- **Power source:** Mains supply.
- **Operating schedule:** Always on.
- **Antenna:** Good (directional) antennas.

These specifications are not limiting in terms of imposing requirements on the design, but rather define the base station as a computationally powerful device with no limitations on power. This allows it to stay on permanently, allowing it to sit dormant, waiting for a signal from the remote unit to initiate communications.

The antenna attached to the base station is likely to be a permanent structure, and therefore may be directional with a high gain, such as a log periodic antenna. These

antennas can typically provide a gain of around 6 dBd. A high gain antenna and high transmit power will increase the SNR of the signal received at the base station, offsetting the poor performance of the remote unit's antenna somewhat, and is discussed in the link budget in chapter 5.

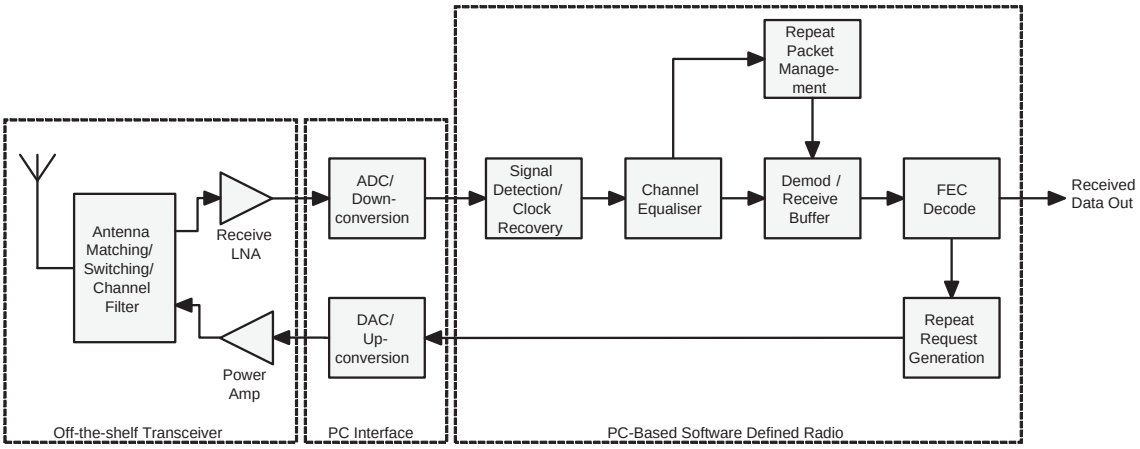


Figure 4.2: Base station functional block diagram.

A functional block diagram of the base station is presented in Figure 4.2 which shows the receive and transmit chains, and describes which modules are implemented as software running on a PC and the RF interface with the PC.

4.3.1 Processing requirements

The assumption of arbitrarily high processing power at the base station is acceptable when designing algorithms to offset the limited resources of the remote unit with the high resources of the base station. It is required, however, to implement the system in practice, so a suitable processing system must be found. To this end it was decided that PC-based software defined radio (SDR) be used. Modern PCs have relatively large processing capacities compared compared to embedded systems, consequently they are a good choice for implementing very computationally intensive algorithms. The base station software and GNU Radio [32], the SDR platform upon which it was developed, is discussed in chapter 11.

In practice, there will be limits to the processing power of the bast station hardware. The semi-blind low density parity check (LDPC) turbo equalisation algorithm especially (discussed in chapter 8) requires a very high amount of processing resource to run. Increasing the processing power results in a shorter length of time required to equalise and decode a received data codeword. As the noise increases and more

iterations are required for this technique, the required processing time increases. Also, as the channel response time modelled by the equaliser increases, the processing time increases geometrically. This is obviously worst on poor channels with a very long response time. This is manifest in the problems that were encountered performing simulations of the equaliser with the ITU-R poor channel, it was impossible to perform these simulations due to running out of resources on the simulation PC. It should be possible to implement the LDPC turbo equaliser in parallel over multiple processor cores, however the mechanism by which this might be realised has not been explored as part of this project.

4.4 The asymmetry of the link

The proposed communications system is specified to be asymmetric in design, with the primary communications link sending data from a low-complexity, battery operated remote unit to a high-complexity base station. There is no requirement for substantial amounts of data to be sent from the base station back to the remote unit. These asymmetries in the power available to each end of the link and the one-directional nature of the link were important when choosing techniques to incorporate into the system.

The high computing power of the base station allows complex algorithms to be implemented at the base station to increase the throughput of the link by reducing the bit error rate and the amount of probe data that is required to be sent with the data payload. This amounts to less time required to be on air for the transmitter to transfer a given amount of data and thus helps conserve its battery life. LDPC coding is a modern FEC technique which provides performance close to a communication channel's Shannon limit. This requires a high complexity decoder to be implemented at the receiver. Quasi-cyclic LDPC (QC-LDPC) codes are a subset of LDPC codes which can be generated at the transmitter side of the link by a simple encoder based on feedback shift registers. These techniques allow for high quality FEC encoding to be applied to the signal sent from the remote unit at minimal computational expense, while minimising errors at the receiver and thus the number of repeat requests required. These techniques are discussed in chapter 7. A further refinement of LDPC which is particularly well suited to difficult channels with fading and multipath, such as typical HF channels, uses prior knowledge of the structure of the LDPC codes to allow for 'blind' estimation of the channel from the received data signal (without

prior knowledge of the channel as can be provided by comparing received probe data with a local copy). This technique, known as blind LDPC turbo equalisation is discussed in chapter 8. It requires a very large amount of processing and memory resources, but the reduction in required probe data required to be sent with the transmitted signal gives tangible improvements to the throughput of the system over typical, contemporary systems as evaluated in chapter 12.

The low complexity of the transmitter unit requires that any error control techniques that are implemented on it be computationally simple enough to fit in the limited processor and memory resources of a low power processor. Chapter 10 describes the implementation of a quasi-cyclic LDPC (QC-LDPC) encoder on a Microchip dsPIC30F6022A DSP microcontroller, which provides performance sufficient to implement a 2 kb/s link with a processor clock speed of 64 MHz (which can be further increased on this processor). The system evaluation in chapter 12 shows that the error correction performance of the QC-LDPC code is very high, especially considering the low complexity of the hardware used to implement it. The technique used is fully compatible with the high performance blind LDPC turbo equaliser implemented on the base station.

Additionally to the processing capacities, the power requirements of the transmitter unit make it desirable that transmissions are kept as short as possible. Factors affecting the time taken to transmit a given length of data through a communications system include;

- The signal data rate,
- The redundancy added to a signal during forward error checking encoding,
- The time taken to service repeat requests from the receiver,
- The length of supplementary signals added to the transmitted signal, such as training data and packet structure.

It is desirable to minimise the latter three items, and increase the data rate of the signal to minimise the on-air time of the remote unit. There are a number of dependencies within the items. For example, as the coding rate increases (and less redundancy is added to the signal) the data rate of the transmitted data also increases but at the expense of noise performance, which may entail a large number of repeat requests from the receiver to be serviced, again reducing the performance of the system.

4.5 Locations

There were no predefined specifications given regarding the geographic locations of the remote unit or base station. The location will, however, have an impact on the performance of the devices for a few reasons.

4.5.1 Latitude

The ionosphere has different properties at different latitudes, with channel conditions typically becoming worse at higher latitudes [14]. Various studies have been performed to characterise the HF channel at high latitude, notably the DAMSON (Doppler and multipath sounding network) project [33]. At auroral latitudes, the Doppler spread of an ionospheric HF channel can become very high, reaching the order of 10 Hz [14][34]. Doppler spread of this magnitude becomes very difficult to compensate for, and requires special consideration to be given to the design of the error control coding, modulation and equaliser algorithms used in the communications devices. This is due to frequent fades of the received signal and fast, random changes to its phase and frequency. If the devices are to be used at high latitudes, a robust modulation technique such as BPSK or DQPSK (differential quadrature PSK) will most likely be required to encode the signal. High Doppler shift at these latitudes will cause problems with OFDM techniques, breaking the orthogonality between subcarriers and causing interference between them [12]. Additionally there may be issues implementing the turbo equaliser techniques described in chapter 8, as the simulations performed in chapter 12 show poor performance on channels with wider Doppler spread. A robust waveform designed for use in poor quality channels such as STANAG 4415 [24], may be the best choice for implementing high latitude digital communications.

At lower and mid latitudes, nearer the equator, the HF channel conditions are typically better, with lower Doppler spread and shift [14]. The techniques discussed and tested in chapters 8 and 12 are probably more applicable to these situations, and should improve throughput and reduce the energy required to send a given block of data.

4.6 Scheduling the communications

As the proposed system integrates FEC, ARQ and a complex blind equalisation algorithm which can take a substantial length of time to complete, a means for scheduling the transmissions from the remote unit must be devised. The remote unit must know when to send a new packet and when to send a repeat. This requires that the base station send acknowledgement packets back to the remote unit, telling it when it has successfully decoded a packet and when it requires further information.

As it is undesirable to leave the remote unit in receive mode for a long time as it waits for an acknowledgement from the base station, which may take a variable length of time up to a few seconds per packet to respond, it may be best to have 'windows of activity' defined a given length of time after the packet has been sent to check for acknowledgements. Then if the base station successfully decodes a packet quickly, it will wait a predefined length of time to send an acknowledgement back to the remote unit.

The case where a packet is not successfully detected by the base station, or an acknowledge is not detected by the remote unit, must also be allowed for. In this case a predetermined number of windows of activity could be used, these will be known to both the remote unit and the bases station. For example, if the remote unit sends a packet that is not detected by the base station, it will wait for n windows of activity before simply resending the entire packet. The number n should be set large enough that if the packet was indeed detected by the base station, that there is a low chance that all of the acknowledges sent back from the base station to the remote unit each time are not detected.

4.7 Channel occupancy

The availability of a HF channel at a given time is dependent upon the atmospheric conditions, but also upon the current spectral occupancy, i.e. the other users communicating on the channel. The occupancy of the HF spectrum is highly dependent upon the geographic location, with western Europe seeing the highest occupancy in the world. Various studies have been carried out to model the occu-

pancy, which provides insight into the power of the man made noise which will be encountered [11][35][36]. The spectral occupancy changes over different times, with evening to night times seeing peak use of the HF spectrum. It would be worth integrating such a model into the proposed communications device prior to deployment, based on observed spectral occupancy Figures of the target area (if such Figures are available). This would allow the communications system knowledge on when the best time to transmit, to minimise man-made noise might be.

4.8 Requirements Conclusion

The above gives a list of requirements and suggestions for consideration prior to deploying the proposed system. It is by no means an exhaustive list, and a further requirements analysis would be required once a better knowledge of the final location of the modems is acquired. The hardware requirements given are sufficient to run the algorithms described later on in this document, and should suffice as a starting point for implementation of communication systems based upon these algorithms. The requirements for the remote unit are based on providing the most data transferred per Joule of battery energy, while the base station requirements mainly specify that as much processing power as possible be requisitioned to implement highly processor intensive algorithms for receiving and decoding data received from the remote unit. These requirements ought to be refined to suit the requirements of any particular applications used in practice.

Chapter 5

Link budget analysis

A link budget analysis is produced to quantify the parameters of a link with a view to defining performance limits of a communications system. Such parameters include required signal to noise ratio (SNR) and the definition of required transmit power and maximum allowable noise (which affects the bandwidth of the receiver, architecture of the amplifiers and components used), maximum data rates, range of the link, antenna gains and frequency.

Due to the unknown nature of the link in which the proposed system will be used, it is difficult to perform a thorough link budget analysis. This chapter therefore describes the various parameters which must be considered when creating a formal link budget. The end of the chapter provides simulation results for a number of links, with parameters chosen to reflect the given specifications of the system, as described and derived in chapter 4.

5.1 Overview of the link

The proposed communications link has been quite broadly defined without many particular requirements. The specifications for the base station are consequently nonrestrictive; section 4.3, which gives the base station requirements, states that the base station will not be a limiting factor in the design of the system and can be assumed to be of high performance, with a good quality, directional log periodic type antenna and mains powered hardware with a high computational capacity for implementing complex algorithms. The remote unit's specifications, detailed in section 4.2

are, however, much more restrictive.

The requirements for the remote unit will be briefly reiterated here. They are a requirement that the unit transmit approximately 3 kB of data per day for at least two weeks (ideally a month or more) with the limited amount of energy available from four alkaline D-cells which corresponds to approximately 15 Ah at 6V DC [29]. The antenna used to transmit this data will be of poor quality in terms of its gain and efficiency, chapter 9 discusses using a buried antenna as a suitable model for the antenna. The 3 kB Figure for data does not include overheads for error correction, which itself includes redundancy from forward error correction (FEC) coding and repeat transmits from the automatic repeat request (ARQ) system. Chapters 7 and 8 discuss FEC and ARQ error control techniques and chapter 12 includes results of bit error rate simulations for LDPC encoded data.

5.2 Path losses

The channel represents the path from the transmitting antenna to the receiving antenna. The transmitted signal is attenuated by the channel with respect to the following losses [7];

- **Ground loss** (L_g) is caused when the signal reflects off the ground in multi-hop systems, and is dependent on the conductivity of the ground. Reflecting off low conductivity surfaces will result in lower losses than high conductivity surfaces such as the sea, for example.

- **Free space propagation loss** (L_{sp}) describes the loss of signal power as the signal travels from the transmitter to the receiver. The signal spreads out as it travels away from the transmitter, the electromagnetic flux and field strength therefore falls as a function of distance and this is manifest as a loss at the receiver. The path loss due to signals propagating through free space is given in terms of the wavelength λ, and path length, D, in dBs by [10]

$$L_{sp} = \left(\frac{\lambda}{4\pi D}\right)^2. \tag{5.1}$$

  which represents the attenuation of a signal due to the combination of dispersion as it passes through free space (which is given linearly as $1/(4\pi D^2)$)

and the effective aperture of an isotropic antenna (which is given linearly as $\lambda^2/(4\pi)$). Assuming a path distance of approximately 4000km and a frequency of 10MHz the propagation loss can be calculated as $L_{sp} \approx 64$ dB.

- **Ionospheric absorption** (L_a) is similar to the ground loss, except that it occurs in the ionosphere and represents the power losses due to a portion the signal passing through or being absorbed by the ionosphere. Absorption is typically strongest through the D-layer, which is the lowest layer of the ionosphere. The losses are proportional to the inverse of the frequency and typically reach 20 dB [37]

- **T**he **fade margin** (L_f) is an extra loss integrated into a link budget to account for variations in the received power of the signal due to fading effects in the channel. This term is added to ensure reliability of the link in the presence of fading.

- **Polarisation mismatch losses** (L_p)are caused by the polarisation of the skywave and the polarisation of the receive antenna not being perfectly matched. As the polarisation of the skywave is liable to change randomly this is difficult to compensate for at the receiver side without using multiple antennas. Polarisation mismatch losses are, on average, 3 dB [37].

The total loss of the link is the sum of all the associated path losses,

$$L_{tot} = L_g + L_{sp} + L_a + L_p. \tag{5.2}$$

Determining these parameters for a long range HF link is non-trivial, involving such factors as the properties of the ionosphere which are affected by the time of day, season, sunspot number, and geographic factors which affect the incident angle of the signal. As these parameters are typically difficult to calculate by hand, computer programs have been written to aid the development of a link budget. One of the more popular of these programs for modelling the HF channel, VOACAP [8], can perform all the requisite calculations to give a good estimate for the path loss. This program is used below to find channel conditions for various paths.

5.3 Noise

The capacity of the channel is defined as the maximum sustainable bit rate achievable through the channel. The equation describing the channel capacity was derived mathematically by Shannon [38];

$$C = B\log_2\left(1 + \frac{S}{N}\right) \tag{5.3}$$

where C is the channel capacity in bps, B is the bandwidth, S is the band limited signal power and N is the band limited noise power. While this limit is theoretically achievable given an optimum channel coding scheme with an infinite block length, in practice this limit can not be achieved. However, recent LDPC coding techniques have been shown to achieve bit error performance approaching this limit [39].

The channel additionally adds noise, which reduces SNR at the receiver. The sources of noise include;

- **Thermal noise** is the radiation that all matter with a temperature above absolute zero emits and is therefore pervasive in all systems. It is the noise that is generated by thermal action within the receiver itself. It can be described by

$$v_t = 10log_{10}(kTB) \tag{5.4}$$

  where v_t is the noise power in dBW, $k = 1.38 \times 10^{-23}\ J/K$ is Boltzmann's constant, T is the temperature in Kelvin and B is the bandwidth within which the noise is measured in Hertz.

- **Man made noise** which is radiated by electronic devices such as spark gaps in car engines and power lines. This includes other RF communications devices. Although these are generally limited to a particular band, or channel, of the electromagnetic spectrum, they typically radiate a small amount of out-of-band power which can affect users using nearby frequencies. The receiver itself uses a band pass filter to select the channel of interest. As no filter is perfect, noise from signals in nearby bands will pass into the receiver. A complex, filter with a high pole number is required to mitigate this noise.

- **Atmospheric noise** is generated by electrical storms in the atmosphere, and will vary according to the weather in the vicinity of the link path.

- **Galactic noise** is generated by the Sun and other sources of radiation originating from outside our planet.

The magnitude of man made, atmospheric and galactic noise (υ_m, υ_a and υ_g respectively) have been measured and documented by the ITU for a range of conditions and frequencies [2]. A graph of noise sources in the HF frequency band is reproduced in Figure 5.1. In this graph the left axis shows the metric F_a, which is the *noise factor* of the channel, that is, the ratio of noise seen at the receiver to the expected thermal noise given a noiseless channel. In equation form,

$$F_a = 10\log_{10}(f_a), \tag{5.5}$$

where $f_a = P_n/(k t_0 B)$, P_n is the noise power, k is Boltzmann's constant, t_0 is the reference temperature of the system (typically taken as 290K) and B is the bandwidth of the signal.

The different noise sources have different properties which affect the capability of the receiver's algorithms to compensate for them. Atmospheric noise has a large impulsive component, being generated predominantly from lightning storms worldwide which emit pulses of wide spectrum interference. Thermal and galactic noise is more homogenous and can better be described or modelled as Gaussian noise sources. Man-made noise includes interference from other users of the RF spectrum. The ensemble of signals from these users will have both impulsive, Gaussian and 'coloured' (where power changes with frequency) noise properties, and is liable to change over time as users start and stop transmitting at different times and in different locations over the course of a day.

5.3.1 Compensating for noise

It is worthwhile mentioning the noise mitigation algorithms employed in the proposed communications system. Two types of error correction will be used, namely, forward error correction (FEC) and automatic repeat request (ARQ). These techniques encode the transmitted data with redundancy to allow successful decoding of noise-corrupted packets at the receiver and send requests back to the transmitter upon unsuccessful receipt of a packet, respectively. Chapters 7 and 8 give a more in depth discussion of the different techniques proposed for use with the system.

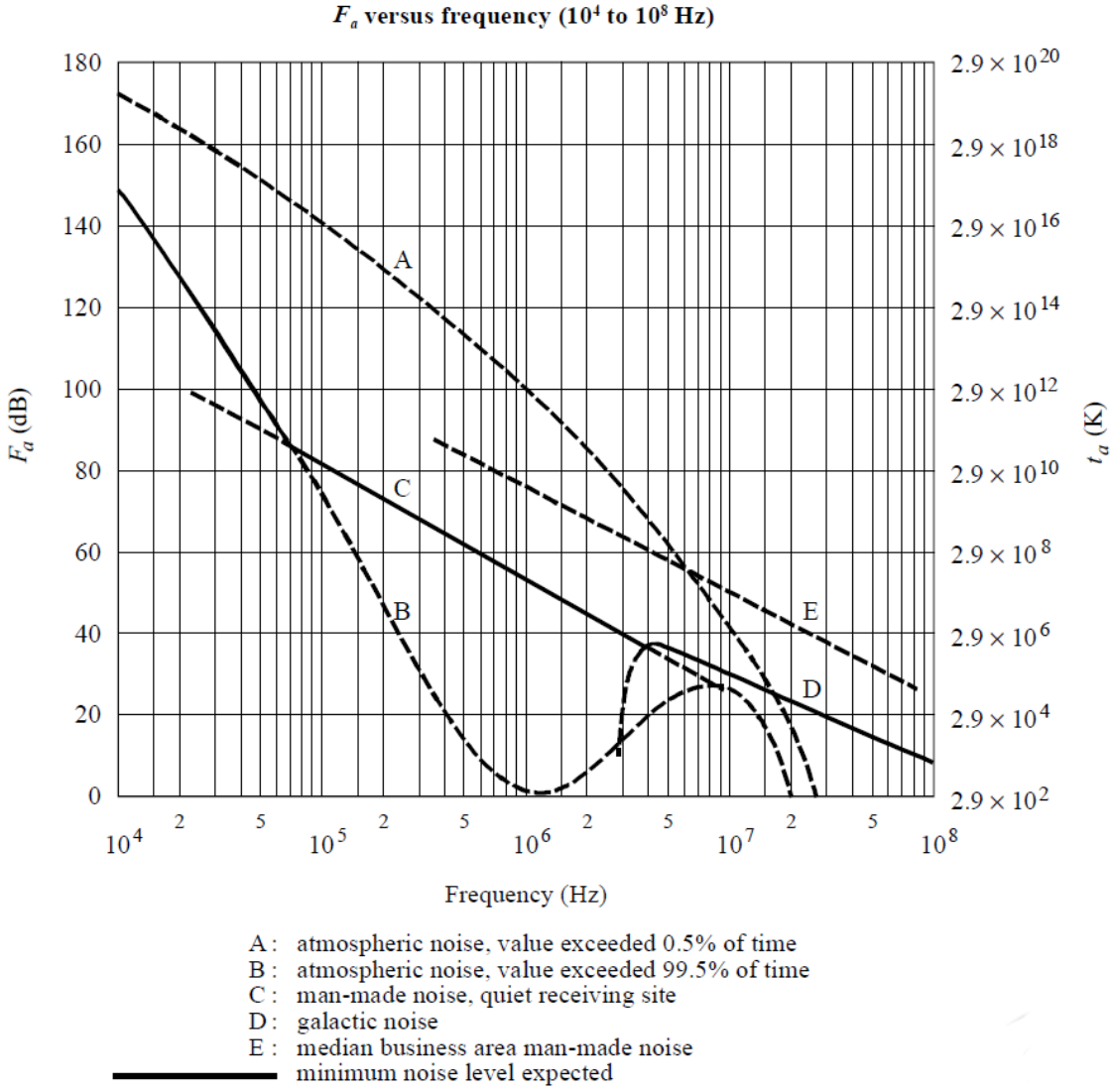


Figure 5.1: Graph of various noise sources in the HF band (from [2])

The code proposed for inclusion in the system is a 2044-bit quasi-cyclic low density parity check (QC-LDPC) code with a codeword length of $n = 2044$. This code performs well as can be seen from the BER tests performed in section 12.2, requiring only 3dB of E_b/N_0 for a low error rate of 10^{-6} (the definition of E_b/N_0, the SNR per bit, is given below in section 5.5).

The channel equaliser is also subject to decreased performance with channel noise. The equalisation technique chosen for use with the system is a semi-blind turbo equaliser, described in section 8.3. The evaluation suggests that a bit error rate of 10^{-2} is observed at an E_b/N_0 of approximately 14 dB (which for BPSK with a code rate of 1/2 translates to a SNR of 11 dB in the bandwidth of the signal) for the ITU-R good channel [4] and an E_b/N_0 of approximately 20 dB for the ITU-R moderate channel. It is assumed that this noise performance could be increased by sending more probe data, section 12.3.4 further discusses this.

5.4 Receiver Hardware

The receiver hardware consists of a high gain receive antenna, band pass filter and a low noise amplifier (LNA). The signal is received and out of band signals are attenuated using a band pass filter. The resulting signal entering the LNA, which can amplify the received signal to the required power for the hardware down the chain, is the combination of the transmitted signal and additive noise from the channel. The ratio of signal received compared to band-limited noise is the signal to noise ratio (SNR), which is expressed in dBs.

The noise Figure of the receiver is dependent upon the combination of the low noise amplifier, band pass filter and universal software radio peripheral (USRP), as the front end electronics for the receiver has not been considered as part of this project. It is difficult to define what this Figure is likely to be. It will however have an effect on the link budget and should be taken into consideration in any further, more formal link budget.

5.5 Bandwidth considerations

From the constraints identified above and equation 5.6, the signal bandwidth is seen to have an effect on the received signal. When increasing the bandwidth, the transmit power must also be proportionally increased to maintain the same SNR at the transmitter, if the signal power is not increased then the SNR falls due to the band limited noise power increasing. It is obvious that a higher bitrate entails a higher signal power for maintaining the same SNR as the bandwidth of the signal is increased.

Two proposed modulation techniques, BPSK and QPSK, are known to have null-to-null bandwidths of $2f_b$ and f_b respectively [3]. These Figures assume that perfect raised-cosine filtering is used at the transmitter output. It is possible to transmit twice the data in the same time and signal bandwidth with QPSK as it is with BPSK. The BER curves from Figure 12.2 in the evaluation chapter 12 are given in terms of E_b/N_0 (the SNR per data bit before FEC coding), a normalised noise figure related to SNR by

$$E_b/N_0 = (\text{SNR}).B/f_b \tag{5.6}$$

where E_b is the energy of one bit of data, N_0 is the noise spectral density and B is the signal bandwidth. The energy per symbol bit is given by the following equation.

$$E_b = \frac{E_s}{n_m R_c} \tag{5.7}$$

Where E_s is the received energy per symbol, n_b is the number of bits modulated onto one symbol and R_c is the code rate. In terms of the energy requirements of transmitting a given number of bits via these two techniques, they are equivalent; BPSK will take twice as long to send data at a given symbol rate, but the E_b/N_0 Figure is doubled compared to QPSK meaning that BPSK need only be sent at half the power, alternatively BPSK can be sent at twice the symbol rate of QPSK. Spectral efficiency is the main difference between the two techniques. Given a band limited channel for communications, QPSK is generally a better choice as it uses the available bandwidth more efficiency.

Another consideration with regards to bandwidth is the width of the receiver's channel filter. This is a band-pass filter which ideally should be as narrow as possible to maximally attenuate any out-of-band noise sources. Noise is typically given in the units dB/Hz, representing the noise power per Hz of bandwidth seen at the receiver. This is also apparent from equation 5.6. By decreasing the bandwidth of the filter,

and by increasing the number of poles (and therefore the roll-off), the received SNR is increased resulting in a more sensitive receiver. Filters with a large number of poles generally require special design considerations such as splitting into multiple stages where each stage is independently grounded and shielded.

5.5.1 Simulation of the channel on Matlab

A simple Matlab model was written to simulate the SNR at the receiver for various communications link configurations.

Figure 5.2 shows the results of a simulation which plots the expected SNR observed at the receiver against the background noise Figure (-140 dBW/Hz represents a noisy, industrial environment and -160 dBW/Hz is representative of a very quiet environment [2]). The channel simulated was a 100 km path of 3kHz bandwidth. The transmitter power was set at 5 W (37 dBm), and the transmit and receive antenna gains were both set at 0 dBi.

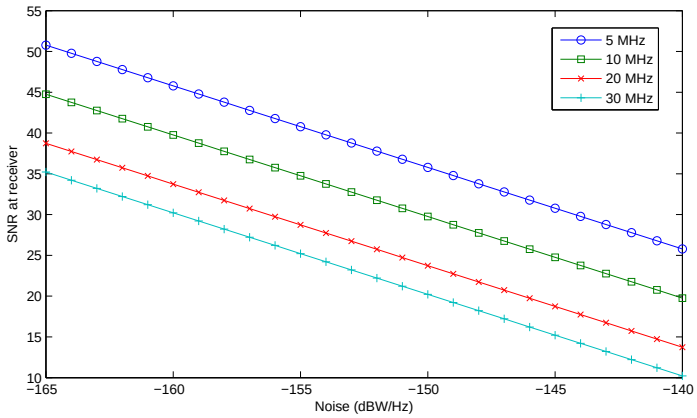


Figure 5.2: Graph of received SNR variation with noise for a 100 km path

Figure 5.3 shows the results of a simulation which plots received SNR versus path length. The parameters are as above and the noise is set at -150 dBW/Hz for all the simulations.

Figure 5.4 shows the result of a simulation which plots received SNR versus bandwidth for a channel of the same parameters as the previous simulations (with a 100 km path length and noise power of -150 dBW/Hz).

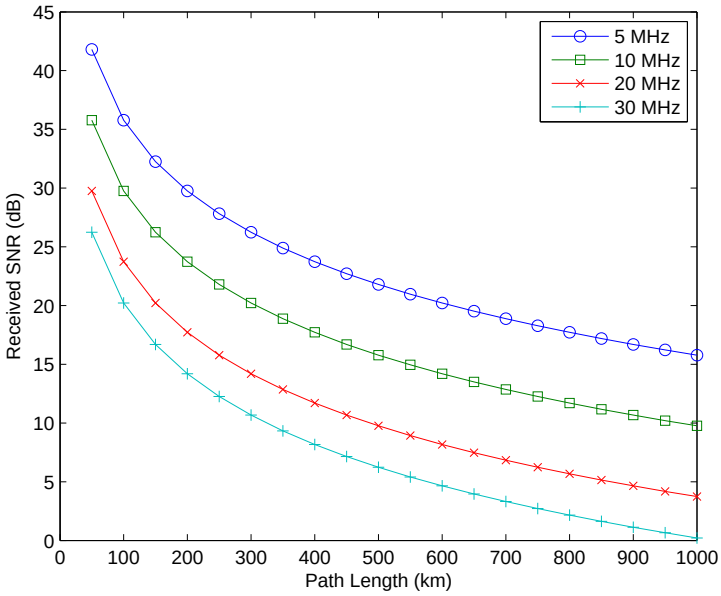


Figure 5.3: Graph of received SNR variation with path length

The above Figures show the typical performance Figures achievable with a 5 W transmitter at various bit rates (which correspond to bandwidth). It is important to remember that these Figures do not take antenna gain or attenuation due to fading into consideration, which must be accounted for in the process of developing a formal link budget.

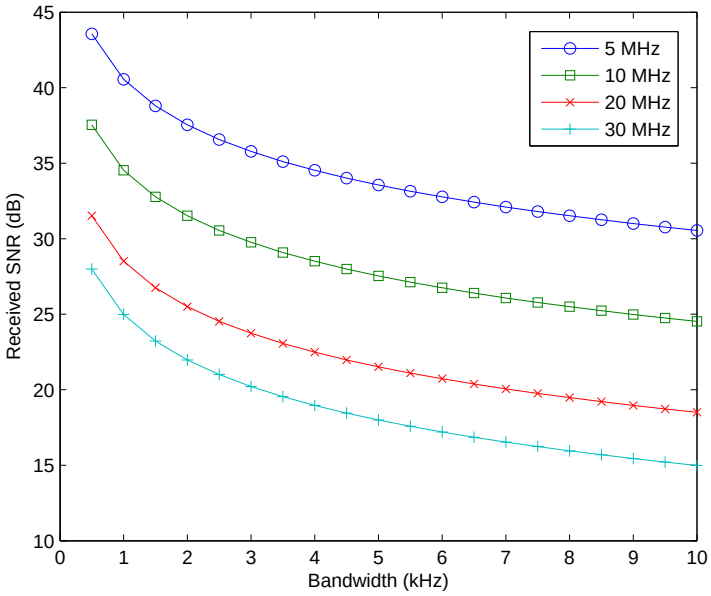


Figure 5.4: Graph of received SNR variation with channel bandwidth

5.6 Link availability

The properties of the ionosphere (such as its height and free electron density) change over the course of a day, and over the longer term, as the seasons and sunspot index change. Due to this, there may only be a few windows of opportunity for successfully communicating at a given radio frequency over the course of a day. For communications systems that are required to be available all the time, this requires operation over a wide range of frequencies, such that a suitable frequency for trans-ionospheric communication can be chosen at any time. For the proposed system, however, which is required to send a small amount of data infrequently (for example once every day), this is of less concern. It can wait until the communications over the operating frequency is possible and transmit at only those times. Due to the limitations of the antenna (as described in chapter 9), there will likely only be a narrow range of frequencies which it is possible to transmit at with practicable efficiency.

There are a few ways in which opportunistic transmission of radio signals could be accomplished. The system could keep a database of suitable times resident on its internal memory, and use that to inform it when to transmit. The suitable times could be generated prior to the remote unit being deployed using HF channel propagation prediction software such as VOACAP (Voice of America Communications Analysis and Prediction program) [8] or REC533 [40] (based on the ITU-R recommendation P.533 [41]). VOACAP predictions of the maximum usable frequency (MUF) have been recently verified to be accurate to within a few MHz for mid-latitudes [42]. Alternative techniques have also been proposed for predicting propagation at high latitudes [43] and using the extraordinary wave [44]. This option is attractive due to its low power requirements. The channel availability data could be used to program a watchdog timer to activate the remote unit only when the channel is likely to be available.

There are two other options which rely more on a ‘trial-and-error’ technique, where either the remote unit could transmit data and listen for a response, or the base station could transmit data constantly and the remote unit could listen for it. The first of these options, with the remote unit transmitting, is not especially power efficient, due to the power required to transmit requests to the base station then listen for a reply. However it is likely to be more accurate than the second option, where the remote unit listens for requests from the base station. Both of these options are

more accurate than simply guessing based on channel predictions, but require more power to keep the remote unit on-air either transmitting or listening.

In practice, it would probably be most beneficial to combine all three techniques together. Channel predictions could be used to tell both the transmitter and base station when to start sounding the channel. When the channel becomes available, the base station could start sending requests that the remote unit listens for. If the remote unit successfully receives one of these requests, it will then send out a confirmatory message which the base station would then reply to. This describes the beginning of a formal handshake protocol.

5.6.1 Prediction of the channel on VOACAP

VOACAP is a free channel prediction application which is an updated version of the older IONCAP (Ionospheric Communications Analysis and Prediction program) for Voice of America [8]. It has the capacity to estimate the conditions of a HF channel between any two locations on earth, at a given time, date and noise level. It can be used to generate tables of data or graphs, the data from which can include a number of parameters including maximum and minimum useable frequencies.

Four paths were predicted using VOACAP;

- **London to Barcelona** is a path with a length of 1135 km through mid latitudes.
- **London to Stockholm** is a path with a length of 1436 km through high latitudes.
- **London to Manchester** is a path with a length of 263 km representing an NVIS link through mid to high latitudes.
- **London to Rhodes** is a long path with a length of 2794 km through mid latitudes.

Each path was predicted with a log periodic receiver antenna with a gain of 17.15 dBi (which is the default gain of the log-periodic type selected) directed at the transmitter, the transmitter antenna was modelled as an omni-directional with a gain of -30 dBi, which is set low to represent an inefficient, poorly matched antenna. Noise on the channel was set at 155 dBW/Hz which is representative of a quiet rural area. The

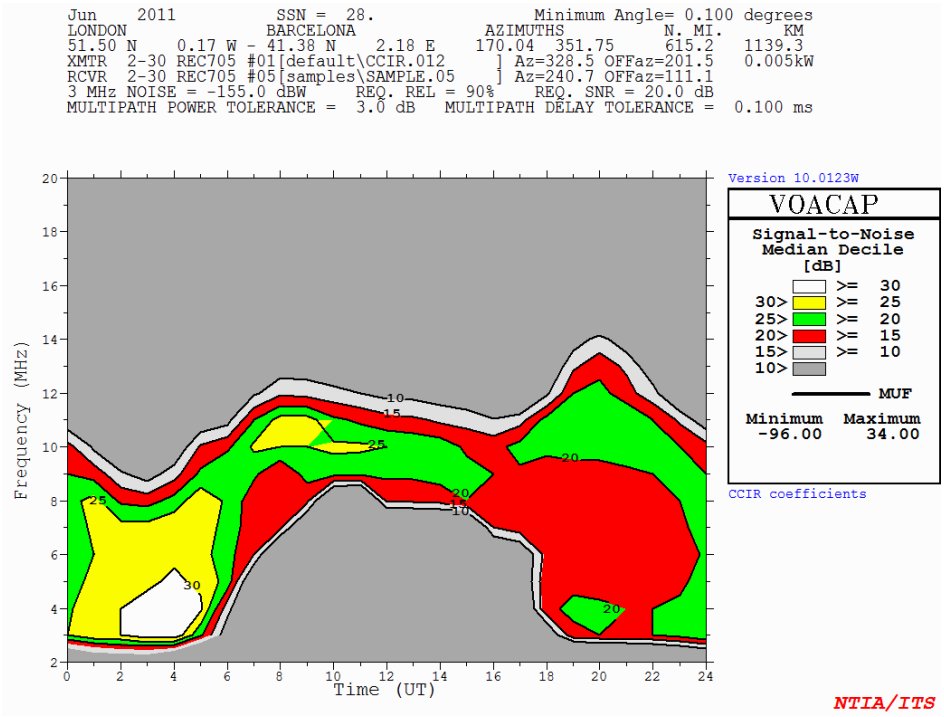


Figure 5.5: VOACAP SNR results for London-Barcelona

transmit power was set to 5 W which is in line with the specifications of the system. Each path was simulated for a summer's day in 2011. For each path, both the expected SNR at various times and various frequencies was found and the expected percentage of days that skywave propagation is possible at a given frequency.

The SNR and percentage of days in which communication is possible at a given frequency are shown respectively for the London to Barcelona path by Figures 5.5 and 5.6, for the London to Stockholm path by Figures 5.7 and 5.8 and the London to Manchester path by Figures 5.9 and 5.8.

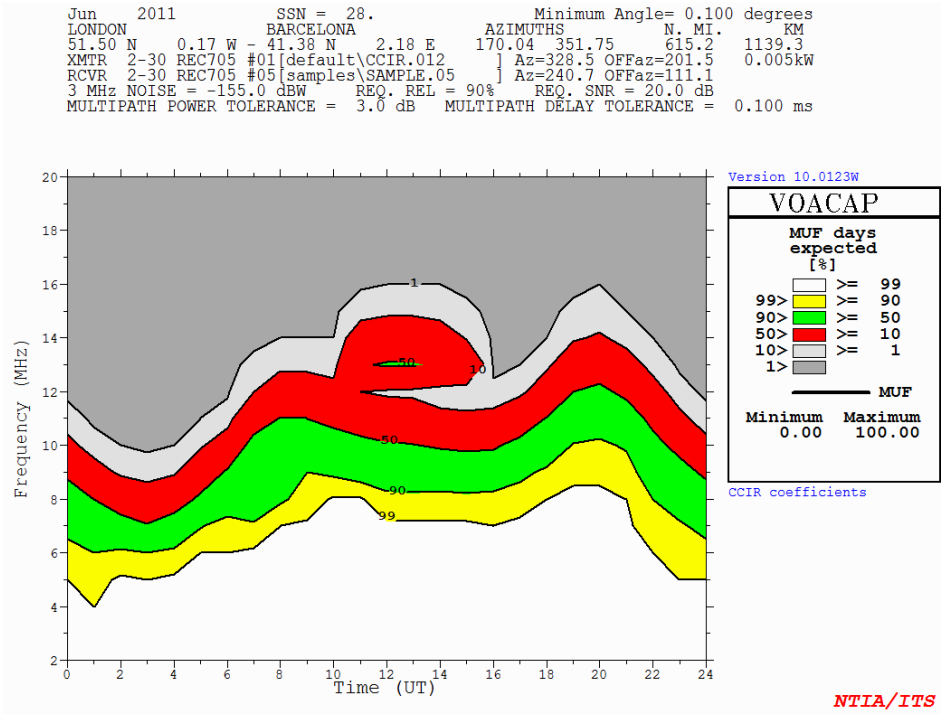


Figure 5.6: Percentage of days with communication possible at MUF for London-Barcelona

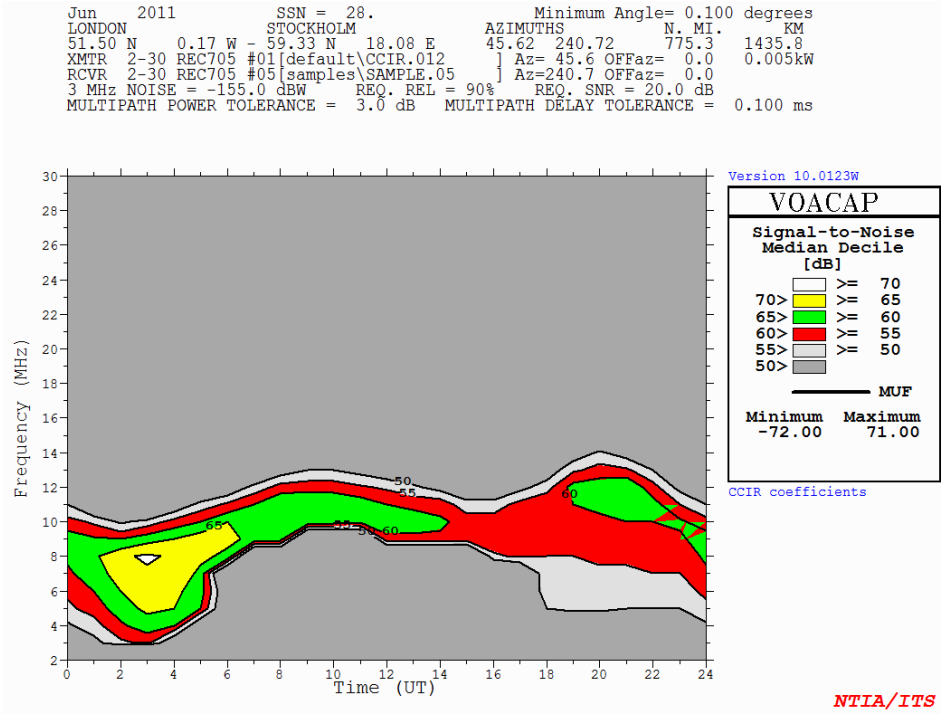


Figure 5.7: VOACAP SNR results for London-Stockholm

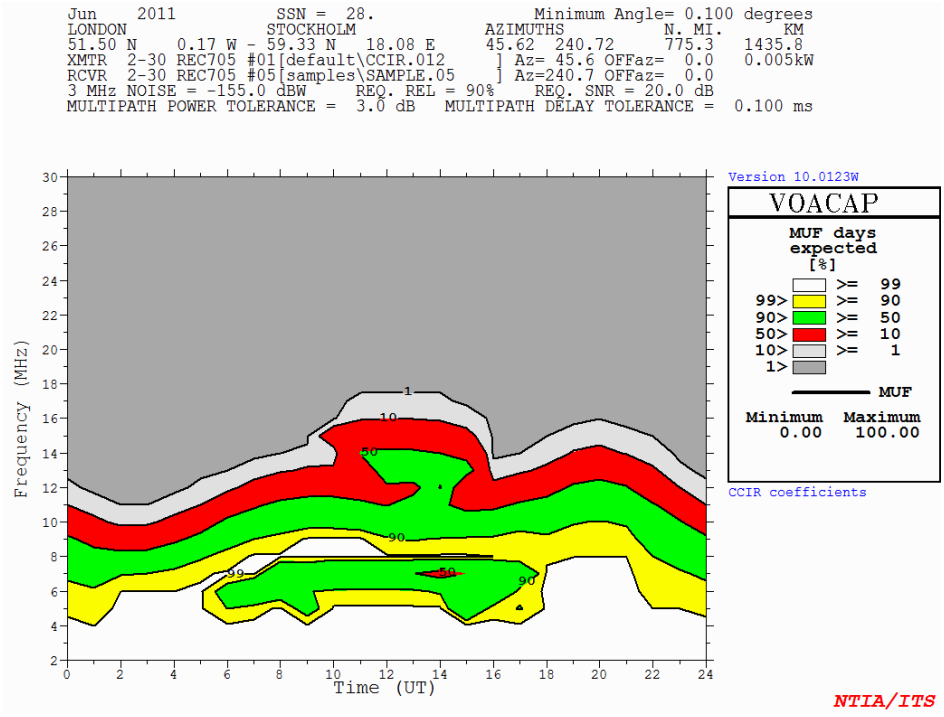


Figure 5.8: Percentage of days with communication possible at MUF for London-Stockholm

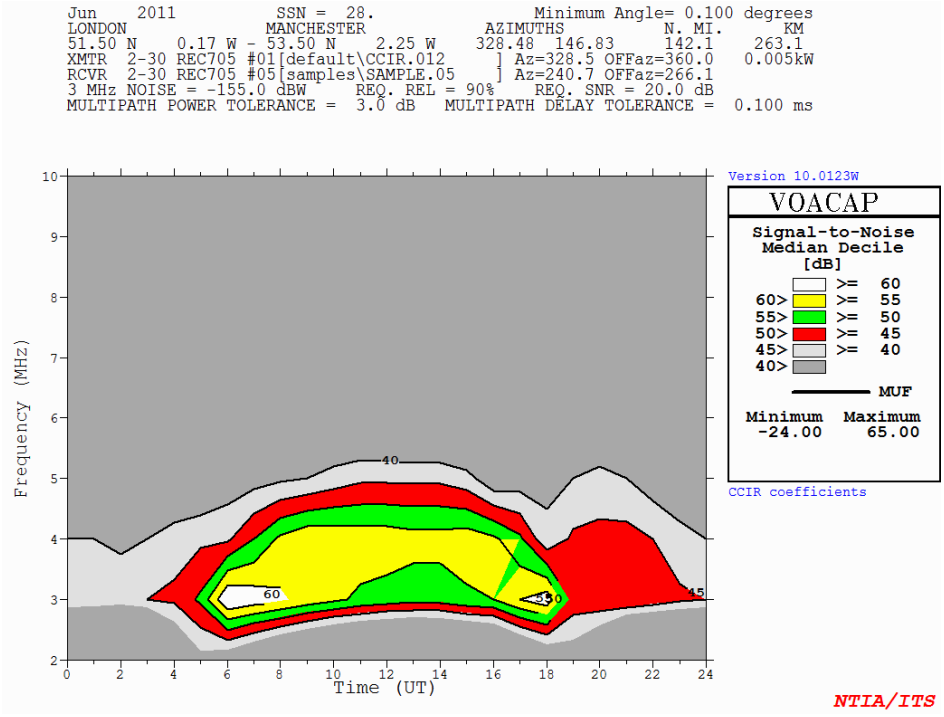


Figure 5.9: VOACAP SNR results for London-Manchester

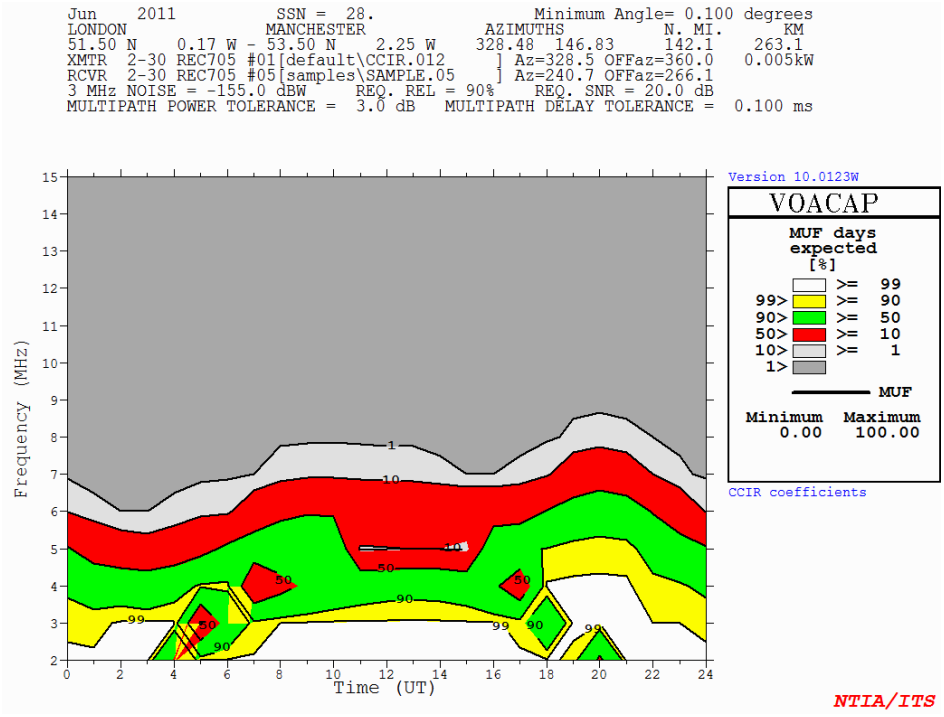


Figure 5.10: Percentage of days with communication possible at MUF for London-Manchester

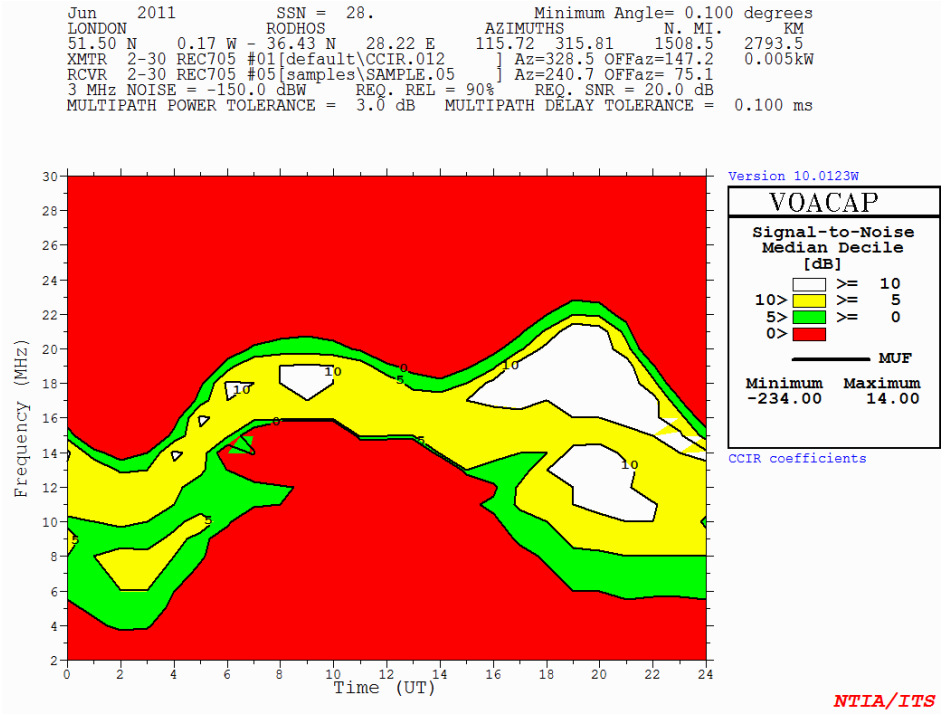


Figure 5.11: VOACAP SNR results for London-Rhodes

It is clear from the above Figures that there is a high degree of variability in the HF channel which is dependent upon path length and geographic location. Given the parameters set out above it appears that observed maximum SNRs throughout the day will vary from 20 dB to 70 dB for a single path. This is sufficient to allow reliable communications using the turbo equalisation technique described in section 8.3 and evaluated in section 12.2. The long path length from London to Rhodes, with the current configuration of poor antenna and low transmit power has too low an SNR at the receiver to be of any practical use in this system.

5.7 Effects of the transmit antenna

The transmit antenna will have an obvious effect on the capabilities of the radio system in the respect that a lower efficiency antenna will have a lower gain, resulting in a lower SNR seen at the receiver. Additionally to this, the takeoff angle of the antenna is likely to have an effect on the quality of the link. It is required that a short distance near vertical incidence skywave (NVIS) link has an antenna with a high takeoff angle of the main radiating lobe. For medium distance paths such as those of approximately 1000 km a main lobe takeoff angle of 10^o to 15^o is preferable. The takeoff angle of buried antennas is further discussed in section 9.4.2.

5.8 Concluding remarks

As the path length and conditions have not been well defined in the specification for this project (see chapter 4), and due to the high variability of possible HF channels (as evidenced by the Matlab and VOACAP simulations performed above), it is difficult to generate a formal link budget. However the results of the simulations presented above give an indication of the expected performance of a 5 W transmit power system over a variety of channels. The higher performance of lower frequencies and shorter path lengths suggest that a 5 W system would be suited to shorter path lengths such as NVIS.

From the results given in section 12.3.3, it is required that at least 20 dB of SNR is seen at the receiver for a BER of less than 10^{-2} using semi-blind LDPC turbo equalisation in the ITU-R good channel. The Matlab simulations show that these

performance requirements should be easily attainable on a real link of 500 km or less. The VOACAP predictions show that these performance Figures are seen on each of the simulated channels, at least at a few hours during the morning and evening hours in each day. For the ITU-R moderate channel, 30 dB is required. The VOACAP predictions show that this Figure may be more difficult to achieve using the poor quality -30 dBi antenna used in the simulations, and the BER will suffer as a result.

The interference from other users of the HF spectrum has not been discussed. This magnitude of interference exhibits a wide range of magnitudes in different continents and at different times of the day, with higher interference typically seen in the evenings over other times of the day, and in Europe over other, less well populated areas [11].

Chapter 6

Modulation techniques for HF communications

To transfer information using a radio communications device, it is required that the information to be transferred is encoded onto a radio signal of the desired frequency. This generally entails the manipulation of one or more carrier waves, which are radio-frequency sinusoidal waves. The information to be transmitted can be analogue, continuous data such as voice, or digitised data.

RF modulation techniques can be broadly classified into two main categories: single-tone techniques use a single local oscillator frequency which is mixed with the baseband signal, and multi-tone techniques which use a number of carrier frequencies (typically regularly spaced over the bandwidth of the transmitted signal). The carriers in multi-tone modulated signals are typically mixed with low bandwidth baseband signals, for each carrier the time taken to transmit a bit is relatively long (the data rate is recovered by transmitting on multiple carriers simultaneously). Conversely, single tone signals are mixed with higher bandwidth baseband signals (comparable to the entire bandwidth of a transmitted multi-tone signal) and it takes a relatively short time to transmit a single data bit. Figure 6.1 shows a conceptual diagram of the differences between single- and multi-tone modulation techniques.

This chapter provides an analysis of the described techniques and a discussion of the reasons that a serial tone modulation technique was chosen for the communications system being developed as part of this project.

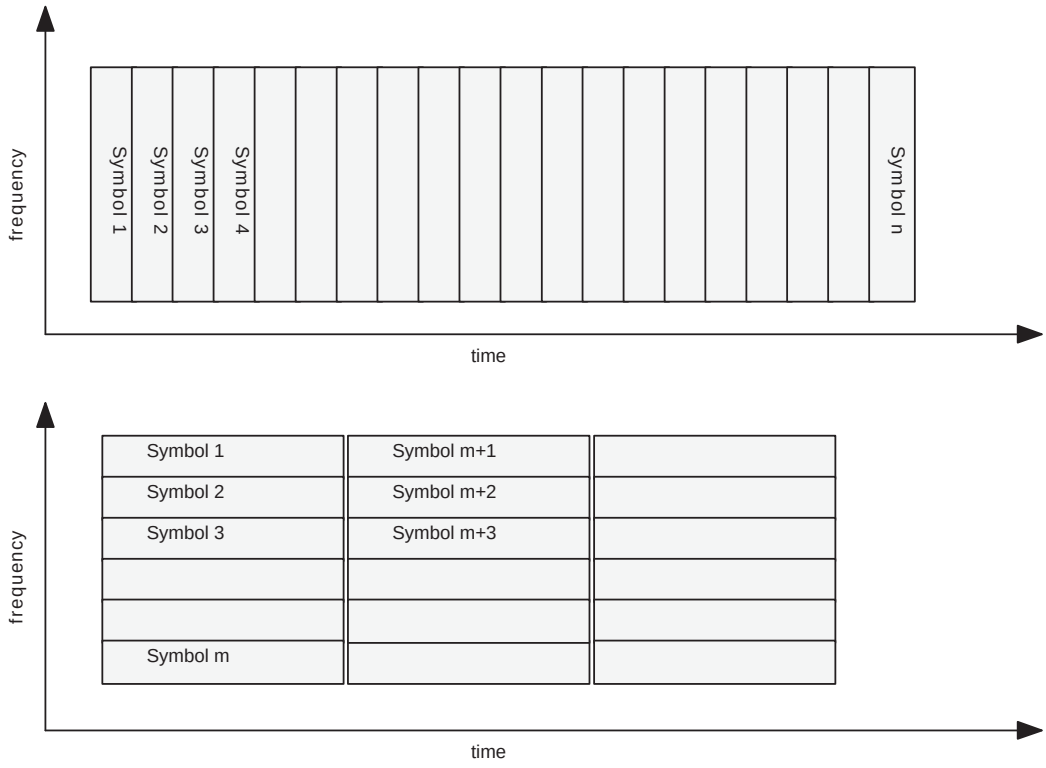


Figure 6.1: Serial tone (top) versus parallel tone (bottom)

6.1 Single tone modulation

The earliest radio techniques, which are still in use prolifically today, involved modulating data onto a carrier wave, which is a locally generated sinusoidal signal at a given radio frequency. Modulating data onto a carrier is performed by changing one or more of its properties, such as amplitude, phase or frequency, this can be achieved by multiplying the carrier wave with symbols which are defined by the data it is required to transmit. A 'symbol' is nomenclature used to describe the transformation the radio signal undergoes to represent a given pattern of data bits. Symbols can include real values such as amplitudes, complex values which can change carrier phase and/or amplitude, or sine waves which, when multiplied with the carrier shift its frequency.

A single tone system is a system which uses one such carrier wave, modulating all of the data to be transmitted onto it, this is in contrast to multi-carrier systems which shall be discussed in section 6.2. The total bandwidth of the signal is then entirely determined by the data modulated onto that one carrier, and the modulation technique used.

6.1.1 Phase shift keying

Phase shift keying (PSK) is a single-tone modulation technique whereby data bits are represented by the phase of the carrier wave. In its simplest form, binary PSK (BPSK), a 0-bit is represented by transmitting the basic sine wave (0^o phase shift) and a 1-bit is represented by transmitting the sine wave in anti-phase (180^o phase shift). By multiplying the resultant wave with a phase coherent copy of the carrier at the receiver, the original data can be recovered.

Higher-order types of phase shift keying are also possible, the most common types are quadrature PSK (QPSK) and 8-PSK, which transmit 2 and 3 bits per symbol respectively. For these techniques each symbol is represented by a different phase of the carrier, and the symbols are generally equally distributed in phase. So, for example, QPSK uses the symbol set $\{0^o, 90^o, 180^o, -90^o\}$ which can also be represented by the set of complex numbers $\{1, j, -1, -j\}$.

Differential phase shift keying (DPSK) is a modification of the above described technique whereby data is transferred based on the phase change of two sequential symbols. The data capacity of such a signal is the same as its non differential equivalent, but noise performance decreases. There is a 3dB increase in required SNR at the receiver to ensure the same error rate as with the equivalent non-differential PSK technique. This is due to noise being found on both symbols instead of one, thus doubling its effective amplitude at the demodulator. DPSK is useful for channels with large amounts of phase distortion, due to the invariance of the technique to common phase distortions between subsequent symbols. This makes it a good choice for poor HF channels.

As phase shift keyed symbols only modulate information onto the phase of the carrier, the technique is invariant to amplitude changes caused by poor channel conditions or noise. This makes it a suitable choice for HF channels, where wide Doppler spread causes frequent fades and rapid changes in the amplitude of a received signal.

6.1.2 Quadrature amplitude modulation

Quadrature amplitude modulation (QAM) is similar to the higher order PSK techniques described above, the difference is that the amplitude is also used to encode

data. QAM constellations, for this reason, can be a lot larger and can transfer many more bits of data per symbol than PSK techniques. For example, 64-QAM, which can be used in good HF channel conditions [20] can transmit 8 bits per symbol.

Quadrature amplitude modulation is not invariant to amplitude variation, unlike PSK. This makes it less suitable for poor channels or low SNR conditions than PSK techniques. High order QAM signals also possess a higher peak to average noise ratio than lower order techniques which has ramifications for the power requirements of the receiver as described in section 6.4.

6.1.3 Gray coding

Gray coding is a method for mapping symbol values to symbols which maps similar data to close symbols [3]. If symbols are adjacent, or otherwise as close as possible, the Hamming distance between the digital values of these symbols is minimised (ideally adjacent symbols' values would have a Hamming distance of only one). The reason for this is that if a symbol is demodulated incorrectly, it is most likely to be incorrectly detected as a symbol close to the correct one. By minimising the Hamming distance between these close symbols using Gray coding then bit errors due to incorrectly decoded symbols is minimised.

Gray coding is generally used in practical systems due to the decrease in errors seen when using it. With turbo equalisation techniques however, as described in chapter 8, it is often desirable to map symbol values so that adjacent symbols have values with as *large* a Hamming distance as possible. Turbo equalisation techniques use an iterative algorithm to simultaneously converge upon the most likely sequence of symbols that was sent and decode the codeword represented by this sequence. By maximising the Hamming distance between adjacent symbols this iterative procedure becomes likely to converge upon the most accurate sequence of symbols and therefore the error rate from the decoder is decreased [45].

6.2 Multi-carrier systems

Multi-carrier (or parallel-tone) systems are parallel in operation, they modulate data onto a series of sub-carriers using simple, low bandwidth modulation schemes such

as quadrature phase shift keying (QPSK). The carriers are then sent in ensemble, each symbol then represents the number of bits modulated onto each carrier times the number of carriers. The symbols can then be transmitted at a much lower rate, reducing inter-symbol interference (ISI) on fading multipath channels. Figure 6.1 shows bandwidth over time characteristics of symbols in conventional single carrier (single-tone) and parallel-tone systems. The serial tone system sends symbols sequentially in time, in this case the bandwidth of each symbol is equal to the total signal bandwidth. Parallel tone systems, however, send blocks of symbols sequentially in time, where each block consists of multiple symbols arranged in the frequency domain. The sum of the bandwidth of every symbol in the block is equal to the total bandwidth of the signal and each symbol is transmitted for a correspondingly longer time to maintain the same data rate.

In a conventional parallel-tone system, it is generally required that a band pass filter be implemented to guard each sub-carrier against crosstalk from the other carriers, if a large number of carriers are desired, this complicates the modem design.

6.3 Orthogonal Frequency Division Multiplexing

Orthogonal frequency division multiplexing (OFDM) [46] is a multi-tone technique which chooses sub carrier frequencies $f_i = f_c + i\Delta f_0$ spaced $\Delta f_0 = k/T_s$ from one another, where k is an integer (normally 1) and T_s is the OFDM symbol duration. A message vector $\mathbf{X}[\mathbf{n}]$ is modulated into symbols using QPSK (or any other simple modulation technique), resulting in a set of channel symbols, which would be transmitted sequentially in a conventional continuous wave system. The signal is then split into blocks of M symbols, which are zero-padded and fed into an N-point inverse fast Fourier transform (IFFT), resulting in an OFDM symbol $\mathbf{x}[\mathbf{n}]$ which when serialised represents a signal of frequency components given by the equation [46]:

$$x[n] = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} X[i] e^{j2\pi ni/N}, 0 \leq n \leq N-1 \tag{6.1}$$

from which the baseband carrier frequencies can be found to be

$$f_i(f) = \frac{i}{NT_s} \tag{6.2}$$

where T_s is the sampling time of the IFFT.

A cyclic prefix is added to each OFDM symbol which consists of its last μ samples, where $T_s\mu$ is chosen to be longer than the length of the channel's impulse response. This allows previously transmitted signals to fully propagate through and clear the channel before sending the next, therefore preventing any inter-symbol interference between sequentially transmitted OFDM symbols [46].

To receive an OFDM signal, an N-point IFFT is performed on the incoming symbols. This rebuilds the original block of sent QPSK symbols which can be demodulated as usual.

6.3.1 Advantages of using OFDM

The use of OFDM entails many benefits. Channel equalisation is very simple as the cyclic prefix added to the OFDM symbols mitigates any sequential inter-symbol interference caused by a long channel response time, requiring only a single tap equaliser (to correct for phase and amplitude) at the receiver [46]. Spectral efficiency is high, approaching the Nyquist rate for a given bandwidth (due to the overlapping spectra of the sub-carriers), and the spectrum of an OFDM signal is very flat, presenting interference similar to white noise to other users of the channel.

OFDM is known to be an effective technique for use in HF systems, providing simplified system and equaliser design and good data throughput rates. This is evidenced by the fact that it was chosen for inclusion as part of the Digital Radio Mondaile (DRM) specification [6].

6.3.2 Peak to Average Power Ratio

The peak to average power ratio (PAPR) is the ratio of the maximum power of a signal to its average power, it is an important attribute as it affects the efficiency and performance of transmitter and receiver equipment. A low PAPR is desirable as it allows the use very power efficient amplifiers [47][48]. High PAPR signals require back-off to ensure linear amplification, which reduces transmitter efficiency [49]. Also the high dynamic range of high PAPR signals pose problems at the receiver's ADC, which is required to have a high dynamic range, increasing receiver complexity.

As an OFDM signal is the sum of a number of independent carrier waves, constructive and destructive interference between them will cause high PAPR of the output signal. It also follows that the PAPR increases as the number of carriers increases, placing a limit on the amount of data it is possible to encode on one symbol. This is the principal drawback to using OFDM, and is especially relevant in a low power radio system due to the requirements of keeping such signals linear in output power amplifiers.

Numerous different techniques have been developed for reducing the PAPR of an OFDM signal. Block coding techniques have been developed which use a subset of the available OFDM symbols with the lowest PAPR as codewords [50][51]. Wilkinson [50] suggests a computer search for appropriate codewords by examining the PAPR of every possible OFDM symbol and discarding all symbols above a desired threshold. Van Nee [51] expands on this idea, but generates codewords by finding complementary sequences, which while not as optimal as a full computer search is a lot quicker. He also examines the minimum distance of his generated code set, showing that a 1/2 rate code will provide a processing gain of 3 dB. Neither technique provides an adequate solution to finding sets of codewords for large numbers of carriers, where computer searches become too long to be practical, Van Nee suggests splitting the OFDM carriers into multiple sets of fewer number and applying coding to each set [51]. This compromise however increases the code overhead and decreases the coding gain.

Performing phase rotation on subsets of the carriers has also been examined. Friese proposes a transmitter scheme where an optimizing iterative algorithm is used to phase rotate subsets of carriers to provide a PAPR arbitrarily close to the theoretical minimum [52]. The PAPR improvement is however, again offset by a cost in overhead, as a number of carriers must be used to declare the phase rotation of the blocks to a receiver. The algorithm itself is computationally intensive at the transmitter, which makes it quite unsuitable for this project, where transmitter complexity must be kept to a minimum with arbitrarily high complexity allowed at the receiver.

Thompson suggested a method of generating an OFDM signal with constant envelope, that is, with a PAPR of 0 dB [53]. Constant envelope OFDM (CE-OFDM) as it is called, feeds the output of a conventional OFDM modulator through a phase modulator, resulting in a constant envelope signal. This requires that the complex baseband OFDM signal be converted into a real signal by modulating it with, for example, pulse-amplitude modulation before phase modulation.

More recently, Slimane [54] has suggested precoding the message vector to reduce its PAPR when OFDM modulated. The theory is presented here. An $N \times L$ precoding transformation matrix, $\mathbf{P}$, is defined to transform a length N message vector $\mathbf{X}_m$ into a length L coded vector $\mathbf{Y}$ that when modulated with OFDM has a low PAPR. $\mathbf{X}$ and $\mathbf{Y}$ are related by

$$Y_i = \sum_{m=0}^{N-1} p_{i,m} X_m \quad i = 0, 1, ..., L-1 \tag{6.3}$$

and the low-pass equivalent form of the OFDM signal is

$$\begin{aligned} x(t) &= \sum_{i=0}^{L-1} Y_i e^{j2\pi i \frac{t}{T}} \quad 0 \leq t < T \\ &= \sum_{m=0}^{N-1} X_m \left(\sum_{i=0}^{L-1} p_{i,m} e^{j2\pi i \frac{t}{T}} \right) \quad 0 \leq t < T \end{aligned} \tag{6.4}$$

where T is the OFDM block length and the guard interval is ignored. The definition of PAPR is

$$\text{PAPR} = \frac{\max |x(t)|^2}{E\left\{|x(t)|^2\right\}} \tag{6.5}$$

which the transformation is designed to minimise. Combining equations 6.4 and 6.5 gives an upper bound for the PAPR in terms of the elements $p_{i,m}$ of the transformation matrix [54]

$$\text{PAPR}(t) \leq \frac{1}{N} \left(\sum_{m=0}^{N-1} \left| \sum_{i=0}^{L-1} p_{i,m} e^{j2\pi i \frac{t}{T}} \right| \right)^2 . \tag{6.6}$$

Here it is convenient to define a set of N time limited complex functions

$$p_m(t) = \begin{cases} p_{i,m} e^{j2\pi i \frac{t}{T}} & \text{for } 0 \leq t < T \\ 0 & \text{otherwise} \end{cases} \tag{6.7}$$

which reduce equation 6.6 to

$$\text{PAPR}(t) \leq \frac{1}{N} \left(\sum_{m=0}^{N-1} |p_m(t)| \right)^2 \tag{6.8}$$

which shows that the PAPR ratio is related to the sum of set of N functions $p_m(t)$. It is apparent here that this set of functions must be chosen to minimise PAPR, and Slimane suggests using cyclic shifts of a seed function with only one amplitude peak. The cyclic shifts ensure that no amplitude peaks are overlapping within any linear combination of the functions, which results in reduced PAPR when compared

to normal OFDM. The matrix elements are found to be

$$\begin{aligned} p_{i,m} &= pi, 0e^{-j2\pi\frac{im}{N}} \\ &= e^{-j2\pi\frac{im}{N}}\frac{1}{T}\int_0^T p(t)e^{-2\pi i\frac{t}{T}}dt. \end{aligned} \tag{6.9}$$

It is further shown that the matrix **P** is required to be orthogonal to maintain the separability of the N transmitted sub-carrier symbols per block without causing ISI [54]. It is shown that to fulfil this criterion the seed function $p(t)$ must satisfy the Nyquist criterion, such as for example the raised cosine function.

Slimane's technique requires an overhead of $N_p = L - N$ carriers, but has shown great improvements to OFDM PAPR. It also allows the definition of a matrix for any size N, results given in [54] show less than 5 dB PAPR with 512 QPSK modulated carriers and an overhead of $N_p/N = 0.2$.

The PAPR of the transmitted data is obviously a major issue as the remote unit, which is tasked with transmitting data through the main communications link will be battery powered, and therefore using transmit power amplifier back-off is very undesirable.

6.3.3 Effects of Doppler shift on OFDM

Doppler shift has a detrimental effect on OFDM signals as it causes the sub-carriers to lose thier mutual orthogonality, which causes inter-carrier interference and a marked reduction in performance [12][55]. This is obviously an issue when using the HF channel, as Doppler shift is commonly observed in the channel (section 2.3).

6.4 Selection of a modulation technique

The selection of a modulation technique for the finished system is dependent on a number of factors. The system requirements from chapter 4 dictate that the system be composed of a small, battery operated remote unit which will transmit data to a complex, mains powered base station. Data transfer will take place primarily from the remote unit to the base station (but a simple feedback path from the base sta-

tion to the remote unit will also be available). These specifications bring an innate asymmetry to the communications link as described in section 4.4.

It is desired that the signal to be transmitted be as easy as possible to modulate at the remote unit, and that the PAPR of the output signal be as low as possible, to maintain a high efficiency output power amplifier. As discussed above a high PAPR requires a large 'backoff' of the output amplifier to maintain the linearity of an outputted signal, where the backoff is effectively a reduction of the power gain of the amplifier. With a class A linear amplifier, any reduction in gain entails an associated reduction in efficiency due to the bias current which flows at all times through it. Ideally a constant envelope modulation technique would be used, which allows the possibility of using very efficient, but non-linear amplifiers such as class D types.

As the data signal is to be received at the base station, which is of arbitrarily high computational complexity, there is no need to attempt to reduce the complexity of the equaliser. One of the strengths of OFDM is that the equaliser required at the receiver need only be a single-tap type, with low complexity and easy to implement at the receiver. In the context of the requirements of the communications link being designed, however, these advantages are ultimately not required.

For the reasons discussed above, it has been decided that a single-tone modulation technique will be used at the remote unit. Using a phase-modulated technique such as BPSK or QPSK should be robust to interference from noise in the channel and require a relatively low SNR at the receiver, which is advantageous given the transmit power constraints and poor quality of the transmit antenna. As mentioned above, ideally a constant modulus modulation technique will be used for allowing increased efficiencies of the transmitter power amplifier, to this end, BPSK or offset QPSK (OQPSK) [3] are both candidates for inclusion into the final system.

6.4.1 Implications for modem performance

There currently seems to be little consensus within the HF community upon whether single-tone or parallel tone techniques provide higher performance (in terms of maximum sustained bit rate under HF channel conditions). The two techniques seem similar in performance, but with different hardware requirements at the transmitter and receiver. The Digital Radio Mondiale specification [6] uses OFDM primarily due to the fact that equalisation of an OFDM waveform is much simpler and less proces-

sor intensive than a single tone signal equaliser. As one of the main intended uses of DRM is for radio broadcasting, the simplicity of the OFDM equaliser implementation is suitable for the cheap, small and possibly battery operated receivers of such broadcasts. Companies like Harris, however, release hardware based around single tone waveforms. This hardware is generally complex and expensive enough that the implementation of a high complexity, single tone equaliser is not a problem.

Chapter 7

Error control techniques

The reduction of errors in a transmission over a noisy channel can be performed using two main techniques. The most straightforward, *automatic-repeat request* (ARQ), alternatively known as *backwards error correction*, is a process where the transmitted message vector $\mathbf{m}$ has a parity check sequence added, which the receiver uses to validate that the packet received is free of errors. If this is not the case, it issues a repeat request to the transmitter which asks it to send the failed packet again. Obviously, to achieve this in practice a full duplex system is required. The alternative technique, *forward error correction* (FEC), involves a process where the transmitter applies a transformation to the message vector before transmitting (a process called *channel coding*), resulting in a coded message vector $\mathbf{c}$, which serves to add redundancy so that the receiver can rebuild $\mathbf{m}$ error-free even in the presence of errors in $\mathbf{c}$.

7.1 Data Redundancy

To ensure accurate receipt of data sent from a transmitter to a receiver over a noisy channel it is necessary to add redundancy to the data. For a vector of binary data, the noisy channel is liable to flip some of the bits resulting in an incorrect received vector. If there is no redundancy placed on the data then it is impossible to detect or correct these errors in the received vector. To detect these errors it is necessary that extra data be sent, such as parity check bits, that can be used at the receiver to check that the received data is correct. Parity check bits are generally the linear com-

binations of the constituent bits of the message vector. The process is performed at the transmitter and the results are appended to the sent data vector. At the receiver the same equations are performed on the received vector and the results are referenced with the received parity check bits, any differences are indicative of an error in the received vector.

The entropy of a length m vector, $\mathbf{x}$ of symbols with probabilities $\mathbf{p} = p_0, p_1, ..., p_m$ is given by the equation [56]

$$H(\mathbf{x}) = \sum_i p_i \log_2 \frac{1}{p_i}. \tag{7.1}$$

This is a measure of the actual information content of the vector and can be seen to be a maximum when $\forall i : p_i = 0.5$, meaning that every bit within the vector is equally likely to be a one or a zero. A vector of data which satisfies this criterion would be completely uncorrelated, perhaps having been subject to perfect compression. It is apparent from equation 7.1 that as the length of the vector increases, its entropy also increases so long as $\forall i : p_i = 0.5$ still holds. Transmitting a vector through a noisy channel entails an increase in entropy as errors are randomly added to it. If that vector is already of maximum entropy it is impossible for these errors to be corrected. To aid the receiver in recovering the correct information from noisy, received vectors it is therefore necessary for the entropy of the outgoing data to be reduced. By adding redundancy (for example in the form of parity checks) to the outgoing data, the length of the vector can be increased while the entropy of the vector remains the same. If this increase in entropy is less than that which the noisy channel adds it is possible to recover the original data from the corrupt received vector [56].

7.1.1 Shannon's noisy channel coding theorem

Following from the previous section and equation 7.1, if the entropy added by a given channel is known, then the minimum required redundancy to add to a signal can be inferred, resulting in a maximum *channel capacity* which can be defined for the given channel. The channel capacity is a term which defines the maximum sustained data rate through that channel, exclusive of redundancy, which is required to be added to allow error free reception of the transmitted signals.

Claude Shannon mathematically derived the channel capacity of any given channel [38]. The representation of this theory, in the form of a mathematical equation,

is given as [38][56]

$$C = B\log_2\left(1 + \frac{S}{N}\right) \tag{7.2}$$

where C is the channel capacity in bits per second, B is the bandwidth of the signal in Hz and S/N is the linear signal to noise ratio seen at the receiver. The channel capacity is often referred to as the *Shannon limit* of the channel.

The Shannon limit imposes a maximum bit-rate that can be reliably transferred across a channel with given bandwidth and SNR, which it is impossible to exceed. It can therefore be used as a benchmark for testing the performance of error correction techniques. The very best techniques will come close to the maximum channel capacity.

7.1.2 Hamming distance

The *Hamming distance* of two binary numbers is defined as the number of bits difference between them [10]. For instance, for the two 8-bit numbers 01001010 and 11001000 the Hamming distance is two, as there are two bits (bit 7 and bit 1) different between them. The Hamming distance is used as a measure of separability, by quantifying how two binary sequences differ. As a simple example of the application of the Hamming distance to digital communications, consider Gray coding, which differs from binary coding in that two adjacent points in a constellation will only differ by one bit (Hamming distance of one), this reduces the bit error rate of the system.

The Hamming distance is also be used as a means to determine the effectiveness of a forward error correction system. Take, for example, a block-coded error correction system where messages of length k are encoded onto a new set of length n codewords. The set of all possible message vectors has 2^k elements, that is one entry per k-bit binary number and the shortest Hamming distance between any two numbers in the set is 1. Therefore the number of codewords is the same as the number of possible message vectors but uses $n-k$ more bits to encode the data. The code rate is therefore k/n and there is a coding overhead of $n-k$ bits. This overhead is used to make the codes more separable, which entails increasing the Hamming distances between them. Then when a received codeword has few errors, that is, too few to make it closer to a different codeword, the original data can be rebuilt by choosing the codeword that is the closest match; the codeword with the shortest Hamming distance from the received word. For this reason it is clearly desirable to

choose a set of codewords that differ from each other as much as possible. The measure used to define the effectiveness of a set of codewords is the *minimum distance*, which is defined as the minimum Hamming distance between any two of the codeword vectors in a set.

7.2 Convolutional Codes

Convolutional coding is a channel coding technique that outputs multiple modulo-2 convolutions of the input stream, resulting in a higher rate output stream. It works by maintaining a first-in first-out (FIFO) memory buffer of the last M input samples, taking the modulo-2 sum of two or more different sets of taps from this buffer and outputting the result. The sets of taps used are given by n generator polynomial functions in terms of the unit delay variable D. (D is commonly used for the unit delay variable when talking about convolutional codes, in block coding X is more often used.) The optimum polynomials have previously been found by computer search and can be found in tables [3]. Take for example the generator polynomials for a rate 1/2 encoder with degree 3;

$$\begin{aligned} \mathbf{g}_1(D) &= 1 + D + D^2 + D^3 \\ \mathbf{g}_2(D) &= 1 + D + D^3 \end{aligned} \tag{7.3}$$

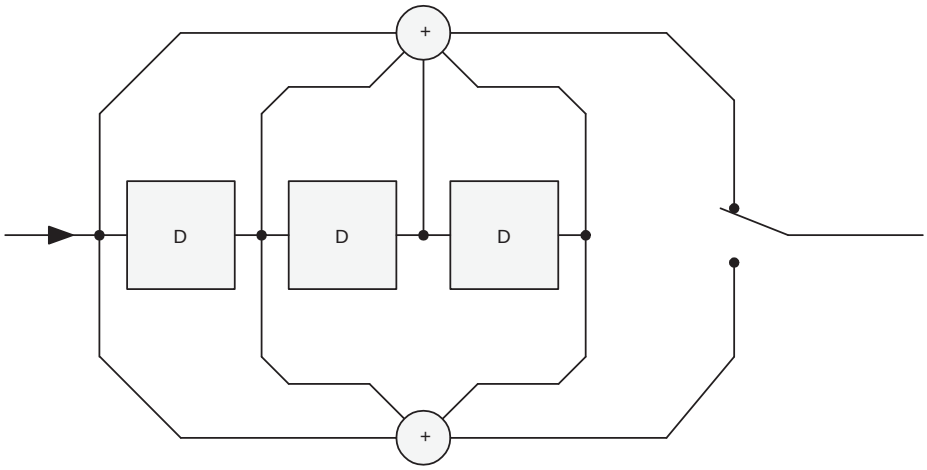


Figure 7.1: 1/2 Rate Convolutional Encoder

Figure 7.1 shows a block diagram of a convolutional encoder that corresponds to the polynomials given in equation 7.3. The message to be encoded comes in from the left into a three stage shift register which is tapped by two sum functions (equivalent

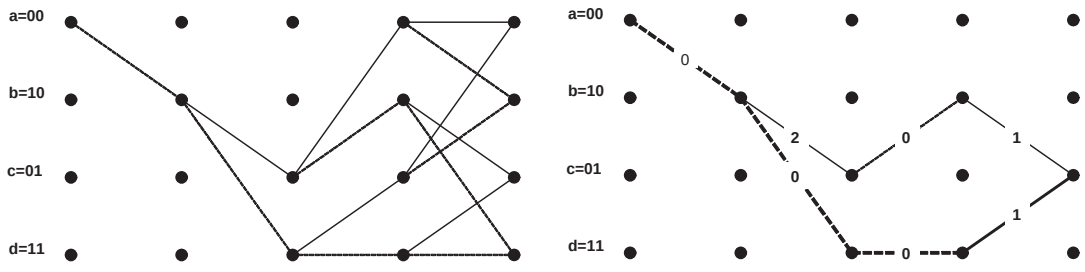
to XOR functions in binary). The output is at twice the rate of the input (assuming one bit is fed in at a time) and it is composed of the concatenation of the two sum functions. The switch is flipped twice every cycle to feed both of the sum functions to the output. The code rate is the number of bits fed into the shift register divided by the number of tap-sum functions.

The *constraint length*, K, of a convolutional code is equal to one more than the degree of the polynomial, and is a measure of how long a particular input bit remains in the shift register. The number of bits shifted into the register every cycle is represented by k. The code rate is k/n; for every k bits shifted into the encoder, one bit of output is generated by each of the n polynomials.

Decoding of a convolutional encoded signal was studied by Viterbi. His algorithm [57] describes a message passing decoder that finds the most likely sequence of input bits that reproduces the received signal (having been corrupted by noise). This algorithm is described below.

7.2.1 The Viterbi Algorithm

The Viterbi algorithm decodes a noisy, convolutional encoded signal by attempting to separate the noise from the signal, using the Markov properties of the encoded signal. A graph can be drawn charting the state of the last two channel outputs over time. Such a graph is known as a *trellis diagram*, due to its similarity with a trellis fence. Figure 7.2(a) gives an example of a trellis. At each step through the trellis a relative likelihood value is assigned to the possible paths that the signal could have taken and unlikely paths are removed from consideration. The path eventually converges to one path of maximum likelihood. Comprehensive discussion of the Viterbi algorithm can be found in [57][10][3].



(a) Trellis diagram (b) Two paths in a trellis diagram

Figure 7.2: Trellis decoding

The Viterbi decoder is an example of a message-passing algorithm. This refers to the way that the most likely path is found. Instead of assessing every possible output for a convolutional encoder and finding the one that best matches the channel output, an iterative procedure is adopted in which messages are passed up each path with the weight metric of that path and its generating input sequence. When two paths cross these messages are compared and the less suitable one is discarded.

Convolutional codes are simple to generate, requiring a shift register and multiplexer. The process of decoding is, however, more complex. As the constraint length increases the length of the paths required to be stored in the decoder increases, which entails a geometrical increase in the number of paths being tracked and compared. The constraint length of a convolutional code is therefore decided by the computational power available at the receiver. This is generally much lower than the codeword length of modern block coding FEC techniques.

7.3 Block Codes

Block codes are another method of encoding redundancy into a stream of message bits whereby the stream is split into blocks of m bits and redundancy is added to each block individually. The transformation from an m-length message vector to an n-length codeword, where $n > m$, is normally performed by a $m \times n$ *generator matrix*, $\mathbf{G}$. The $n-m$ redundant bits are typically parity check bits which are appended to the original message vector to make the codeword. For this reason the generator can usually be split into two sub-matrices; an $m \times m$ identity matrix, $\mathbf{I}_m$, which copies the message vector to the codeword, and a $m \times (n-m)$ sub-matrix, $\mathbf{A}$ which represents each parity check equation to perform on the message vector, which adds the parity check bits to the codeword. The generator matrix can then be defined as $\mathbf{G} = [\mathbf{I}_m|\mathbf{A}]$. To check the received codeword for errors, and to aid in the correction of such errors, it is necessary to define a *parity check matrix*, $\mathbf{H}$, where $\mathbf{G}\mathbf{H}^T = 0$, where $\mathbf{X}^T$ denotes the Hermitian transpose of the matrix $\mathbf{X}$. The parity check matrix can be defined from the components of the generator matrix as $\mathbf{H} = [\mathbf{A}^T|\mathbf{I}_{(n-m)}]$.

One of the earliest block codes invented was the Hamming code, which was a short code capable of detecting up to two errors in a block or correcting one. Further information on this and other simple block codes can be found in [56][3][10].

7.4 Sparse graph codes

Sparse graph codes are a class of channel codes that have seen great success in recent years, with empirical performance levels coming arbitrarily close to the Shannon capacity of a channel. Such codes can be described using sparse bipartate or *Tanner* graphs [58]. A Tanner graph is a graph with two rows of nodes; the bottom row represents codeword bits, and the top represents parity checks. Each codeword bit node connects to a number of parity checks and each parity check node connects to a number of codeword bits. The graphs are called sparse because, in general, the number of connections for each node is far less than the total number of parity check or codeword bit nodes in the graph.

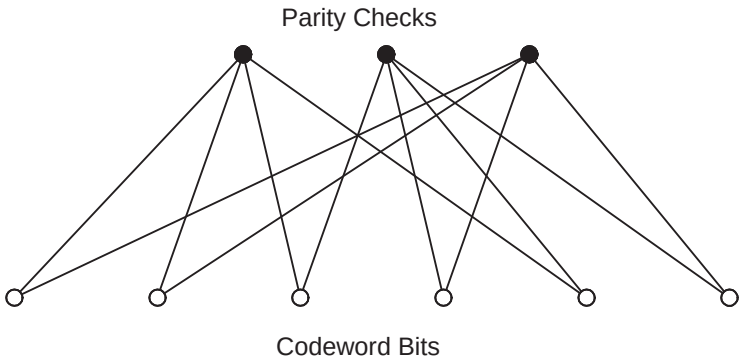


Figure 7.3: Simple example of a tanner graph

Figure 7.3 shows an example of a Tanner graph with six codeword bits and three parity checks. There are four codeword bits in each parity check and each codeword bit participates in two parity checks. This graph is representative of a linear block code with parity check matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{7.4}$$

where each row represents a parity check and each column a codeword bit. It is apparent that the column and row weights (the number of ones in each) correspond to the number of connections per each parity check node (for row weights) and each codeword bit node (for column weights). This example can not truly be called a sparse graph as the number of connections per node is high compared to the total number of nodes and the parity check matrix is clearly not sparse.

Turbo, Low-density parity check (LDPC), repeat-accumulate and fountain codes are all sparse graph codes, the first three families are used on a standard binary symmetric channel whilst the last technique, fountain coding is an interesting code with good performance on the erasure channel. Information on these codes can be found in [56], for the purposes of this project it is only necessary to elaborate on the LDPC family of codes, as these have been selected for use with the proposed communications system for a variety of reasons (see chapter 8). In summary, they have been chosen because they provide high performance, and further techniques can be used to both reduce the computational complexity of the encoder and use the code structure to aid in equalisation at the receiver.

7.5 Low density parity check codes

The low density parity check (LDPC) code was first suggested by Gallagher in his 1963 thesis [59]. At the time of Gallagher's thesis, there was nowhere near sufficient computational power to realise a practical LDPC system, and little further work was performed in the field until the 1990s. The technique uses a very sparse matrix (i.e. composed mostly of zeros) as a parity-check matrix, and hence the name. A construction method of such a matrix is discussed in [59], and has been rediscovered and improved upon by Mackay [39], who suggests a method of semi-randomly generating matrices. The basic algorithm for creating an LDPC generator matrix is as follows: First a very sparse matrix $\mathbf{A}$ is generated by randomly picking n columns of length $n-k$ with t_c ones randomly inserted in each. The variables n and k are the codeword length and message length respectively and t_c is the column weight of the code, that is, how many ones exist in each column. The next step involves swapping the location of bits to try to get the row weights as uniform as possible. If the weight of every row is equal to t_r, where t_r is the average row weight of the matrix, then the resultant code is known as a regular LDPC or Gallagher code. $\mathbf{A}$ can be shown as the concatenation of two matrices

$$\mathbf{A} = [\mathbf{A}_1 | \mathbf{A}_2] \tag{7.5}$$

where $\mathbf{A}_1$ is a $(n-k) \times (n-k)$ square matrix and $\mathbf{A}_2$ is a $(n-k) \times k$ rectangular matrix. Gauss-Jordan elimination [60] is used on $\mathbf{A}$ to get it into the systematic form

$$\mathbf{H}^T = [\mathbf{I}_k | \mathbf{P}] \tag{7.6}$$

where I_k is the $k \times k$ identity matrix, and P is the $k \times (n-k)$ coefficient matrix where $P = [\mathbf{A}_2\mathbf{A}_1^{-1}]$ [39]. The generator matrix is thus

$$\mathbf{H} = \begin{bmatrix} \mathbf{I_k} \\ \mathbf{P}^T \end{bmatrix} = \begin{bmatrix} \mathbf{I_k} \\ \mathbf{A}_2\mathbf{A}_1^{-1\,T} \end{bmatrix}. \tag{7.7}$$

In addition to finding the nearest codeword for a given received, LDPC encoded block, the LDPC decoder also outputs the posterior probabilities of each bit of the codeword. This information is useful for a number of other techniques such as turbo equalisation and hybrid-ARQ (HARQ). These techniques have been explored and are discussed in sections 8.2 and 7.7.1 respectively.

7.5.1 Generating LDPC codes

LDPC codes are generated using a generator matrix, **G**, which is defined as the null space of a very sparse parity check matrix **A**. Many techniques have been proposed for the generation of **A**, including pseudo random [61][59] and algebraic techniques [62]. In general, the design aims are to create a sparse $N \times M$ matrix, where N is the length of the codeword generated when a length $N - M$ input message vector is encoded.

The simplest way of performing this operation, as proposed by Gallager in his original thesis [59], is to randomly select a matrix, ensuring a fixed column weight (or number of ones in each column) of j and a fixed row weight of $i = jN/M$. Row and column reordering is used to ensure the first M rows form an invertible matrix as described above (it is immediately apparent that the example matrix of equation 7.4 is *not* such a matrix). This generates what is known as a regular Gallager code.

The *girth* of a code is an important metric. This is defined as the minimum cycle length in the Tanner graph of the code. A cycle is a path that can be traced from one node, along different paths, back to the original node, as in Figure 7.4 where a length four cycle can be seen emboldened. Short cycle lengths in a code break the assumption of independence that the decoder requires and therefore impair decoding performance (this manifests as an increased error floor). As such, a second step in many random LDPC generation algorithms purges the parity check matrix of any such short cycle lengths.

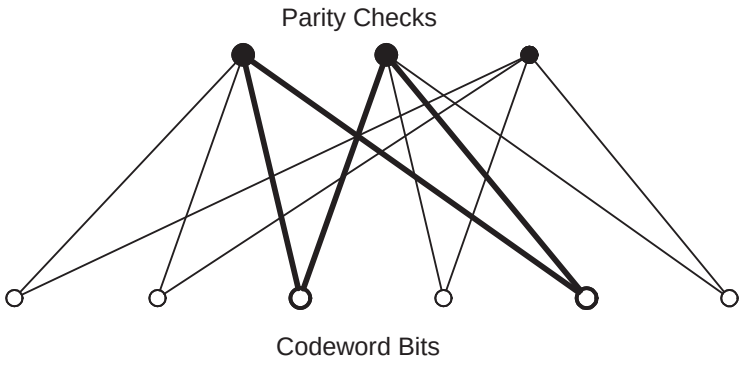


Figure 7.4: A cycle of length 4 (emboldened)

To create the generator matrix, G, the null space of the parity check matrix, A, is found. First, Gaussian elimination and column reordering is performed on the matrix to transform it into *reduced row-echelon form*;

$$\mathbf{H} = [\mathbf{I}_{\mathrm{M}} | \mathbf{P}] \tag{7.8}$$

where $\mathbf{H}$ is the reduced row-echelon form of $\mathbf{A}$, $\mathbf{I}_{\mathrm{M}}$ is the $R \times R$ (where $R = \mathrm{rank}(A)$) identity matrix and $\mathbf{P}$ is an $R \times N - R$ matrix (not necessarily sparse) composed of only zeros and ones. The generator matrix for the code is then given as;

$$\mathbf{G} = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_{\mathrm{N-R}} \end{bmatrix} \tag{7.9}$$

where $\mathbf{I}_{\mathrm{N-R}}$ is the $(N - R) \times (N - R)$ identity matrix. A codeword column vector $\mathbf{c}$ is then generated for in input column vector $\mathbf{m}$ of length $N - R$ by modulo-two multiplication, $\mathbf{c} = \mathbf{G}\mathbf{m} (\mathrm{mod}\ 2)$.

7.5.2 Quasi Cyclic LDPC

There is a class of algebraically constructed LDPC codes known as quasi-cyclic LDPC (QC-LDPC) codes. These codes are constructed from an array of *circulant* matrices over GF(2) (the Galois field of two elements). Their structure allows for a much simpler implementation of the encoder, where the computationally expensive multiplication of the message vector with the generator matrix is replaced with a linear (with N) complexity technique using FSRs [63]. This is obviously of interest to this project, where it is desirable to have as little complexity as possible on the transmitter side (see the remote unit requirements in section 4.2). When invented,

QC-LDPC was thought to have lower performance than randomly generated codes, but subsequent work has shown that well designed QC-LDPC codes in fact perform just as well as them [62].

A circulant matrix is defined as a matrix where any row is the cyclic shift, one place to the right, of the row above and every column is the cyclic shift of the previous column one place down. The identity matrix is therefore a circulant matrix, as each row has a single 1 in it, one cell to the right of the location of the 1 in the row above, and similarly for the columns. Using the $P \times P$ identity matrix $\mathbf{I}_0$ as a basis, a family of circulant matrices, $\mathbf{I}_n$ where $0 \leq n < P$ can be defined as the identity matrix with every row cyclically shifted by n places. For example, with $P = 4$, $\mathbf{I}_1$ and $\mathbf{I}_2$ are given as

$$\mathbf{I}_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} \quad \mathbf{I}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Consider the finite Euclidean geometry E($m, 2^p$), which is an m dimensional space with length down any dimension of 2^p discrete points. This space consists of 2^{pm} points in total, and $q^{m-1}(q^m - 1)/(q - 1)$ possible lines, where any line, $\mathcal{L}$, has 2^{pm} points located on it [64]. If $\mathbf{a}$ is a point on $\mathcal{L}$ then $\mathcal{L}$ is said to pass through point $\mathbf{a}$. Any two lines either have no points in common (i.e. they are parallel) or they have one point in common. If two lines share point $\mathbf{a}$ in common they are said to *intersect* at $\mathbf{a}$. As the number of points in E($m, 2^p$) is equal to 2^{pm}, the same as the number of elements in GF(2^{pm}) (whose elements are $\{\alpha^{-\infty} = 0, \alpha^0 = 1, \alpha^1, \alpha^2, \ldots \alpha^{pm-2}\}$) the Euclidean geometry can be used to completely represent the elements of GF(2^{pm}) [64][65]. Following from this, a line, $\mathcal{L}_1$ that intersects two linearly independent points $\mathbf{a}_0$ and $\mathbf{a}_1$ can be represented as a 2^{pm}-tuple, $\mathcal{L}_1 = \{\mathbf{a}_0, \mathbf{a}_0 + \beta^0\mathbf{a}_1, \mathbf{a}_0 + \beta^1\mathbf{a}_1, \ldots, \mathbf{a}_0 + \beta^{pm-2}\mathbf{a}_1\}$ where β is a primitive element of GF(2^{pm}). It can furthermore be shown that $\alpha\mathcal{L}_1$ is also a line [64]. Another line, $\mathcal{L}_2 = \{\mathbf{a}_0, \mathbf{a}_0 + \beta^0\mathbf{a}_2, \mathbf{a}_0 + \beta^1\mathbf{a}_2, \ldots, \mathbf{a}_0 + \beta^{pm-2}\mathbf{a}_2\}$ is defined, that passes through points $\mathbf{a}_0$ and $\mathbf{a}_2$, and intersects $\mathcal{L}_1$ at point $\mathbf{a}_0$. It is shown in [65] that the two lines $\alpha^i\mathcal{L}_1$ and $\alpha^j\mathcal{L}_2$ can only share one point, $\mathbf{a}_0$ in common, so long as $\mathbf{a}_1$ and $\mathbf{a}_2$ are linearly independent [65]. Each such line defined within this geometry will have exactly 2^p points. The points on these lines represent symbols in GF(2^{pm}) so this implies that the tuples formed by the lines will only share one element in GF(2^{pm}) in common.

Circulant matrices are formed from the tuples defined by the lines in E($m, 2^p$). Each

circulant has a row and column length of 2^{pm}, within which there are 2^p non zero elements, corresponding to the elements of a line. Every row of the circulant matrix, as explained above, is the rotation of the row above it, and therefore each row of the circulant matrix represents another parallel line within the Euclidean geometry. Each possible line can be used to generate a circulant matrix and these can be concatenated to form a row of circulant matrices. The next step is to perform row and column decomposition on each circulant, which shares the non zero elements of the original matrix between new matrices which are concatenated in both directions, until the required column weight is found [63]. The new matrix generated is the QC-LDPC parity check matrix.

7.5.3 Efficient encoding of QC-LDPC codes

One of the strengths of quasi-cyclic LDPC techniques is that it is possible to implement an encoder with much lower processing and memory requirements than that of a normal LDPC decoder by taking advantage of the special structure of the parity check matrix.

To encode a codeword of a conventional random LDPC code (otherwise known as Mackay code), the null space of the parity check matrix forms the generator matrix and this is multiplied with a message vector to form a codeword. While the parity check matrix is itself sparse, the generator matrix, which is of the form given in equation 7.9, consists of an identity matrix and a non-sparse matrix **P**. For large codeword lengths, **P**, which cannot be represented in any sparse matrix format, requires a large amount of memory to store. For example a half rate code of codeword length $N = 2000$ will have a **P** matrix of dimension 1000×1000, requiring at least 1,000,000 bits = 125kB of storage for the generator matrix. This memory requirement is rather high compared to the data memory capacity of most low-cost microcontrollers.

To encode a QC-LDPC code efficiently the previously mentioned FSR-based technique can be used [63]. This technique uses feedback shift registers (FSRs) to encode a QC-LDPC codeword vector directly from the circulant sub-matrices that make up the parity check matrix. As these sub-matrices are circulants only the first row of each needs to be sent, and as the first rows are sparse the memory requirements of the encoder are minimised. To encode a codeword the message is split into sections of l_c bits, where l_c is the length of a circulant sub-matrix. Then the first

rows of each sub-matrix in turn are used to initialise an FSR. The FSR is rotated and the bits of a corresponding message chunk are checked sequentially, every time a bit is detected along the message chunk the FSR is summed with a length l_c register which is initialised to zero at the start. A new register is used for every successive circulant and the codeword can finally be built up from the values contained within these registers. This is a very simplified account of the encoding process which serves only to give an indication of the encoding process, compared to the large matrix multiply which is required for a normal LDPC code.

7.5.4 Decoding LDPC Codes

This section describes the operation of the LDPC decoder. A message vector $\mathbf{m} = [m_1, m_2, ..., m_{N-R}]^T$ is encoded into an LDPC codeword $\mathbf{c} = [c_1, c_2, ...c_N]^T$ (where the notation $\mathbf{A}^T$ is used to denote the Hermitian transpose of $\mathbf{A}$, N is the codeword length and R is the number of parity bits). This code vector is transmitted through an additive white Gaussian noise (AWGN) channel resulting in the corrupt message vector $\mathbf{y} = [y_1, y_2, ..., y_N]^T$.

The aim of an LDPC decoder is to find the value of $\mathbf{c}$ which maximises the likelihood $P(\mathbf{y}|\mathbf{c})$ for a given parity check matrix $\mathbf{H}$, where $\mathbf{Hc} = 0(\text{mod } 2)$. The matrix $\mathbf{H}$ can be expressed as a Tanner graph with edges defined by the placement of 1s within the parity matrix (so, as an example, a 1 at column 5 and row 3 will generate an edge connecting codeword bit node 5 to parity check node 3). Column n, where $0 < n \leq N$, will contain 1s at row indices corresponding to the destination parity check nodes for all edges leading out of codeword bit node n. This set is denoted $\mathcal{M}(n)$ and is the set of parity checks in which check node n participates. Similarly we can observe the placement of 1s in the rows of $\mathbf{H}$ and define sets $\mathcal{N}(m)$ of the codeword bits participating in parity check m.

To decode y, the belief propagation algorithm (otherwise known as the sum-product algorithm) is used, which is equivalent to a maximum likelihood decoder when there are no cycles in the graph defined by A [56]. It is a message passing algorithm that passes probabilities up and down the edges of the graph. Each edge starts out initialised with the received probability of its connected codeword bit. The algorithm then repeatedly examines and updates the likelihood of each codeword bit being a one by first examining the input to each parity check (which must sum to zero in modulo-2 arithmetic by definition), updating the probabilities of each edge to that

which best satisfies the parity check for each edge, then passing these values back to the codeword nodes, where each node updates its probability based on the values received from all connected parity check nodes and the original received data. When the *pseudo-posterior probabilities* $\hat{\mathbf{c}}$ derived from the check nodes satisfy the parity check equation then decoding is declared a success. A very good tutorial clearly explaining this process is found in [66].

7.5.4.1 The belief propagation decoder

The algorithm for decoding LDPC codes as described above is split into an initialization phase, iterative 'horizontal' and 'vertical' steps, finally stopping when the inferred value of the codeword bits satisfies the parity check equation.

Initialization. Prior probabilities of the codeword bits are defined for the probability that codeword bit c_n is equal to zero and one respectively; $p_n^0 = P(x_n = 1)$ and $p_n^1 = P(x_n = 1) = 1 - p_n^0$. These are set to the likelihoods of the received bits. Messages q_{mn}^0 and q_{mn}^1 are set equal to p_n^0 and p_n^1 respectively for all m, n for which $\mathbf{H}_{m,n} = 1$.

Horizontal step. The horizontal step of the algorithm examines the messages received at each parity check node, updating each received edge with a value that reflects what the algorithm believes it should be to satisfy the parity check equation

$$\sum_{n \in \mathcal{N}(m)} \hat{\mathbf{c}}_n = 0 \,(\text{mod } 2). \tag{7.10}$$

To this end, the algorithm looks at every parity check node in turn (indexed $m = [1, 2, \ldots, M]$) and for each $n \in \mathcal{N}(m)$ the probabilities r_{mn}^0 and r_{mn}^1 are defined [56][66]

$$\delta r_{mn} = \prod_{n' \in \mathcal{N}(m) \backslash n} \delta q_{mn'} \tag{7.11}$$

$$r_{mn}^0 = \frac{1}{2}(1 + \delta r_{mn}) \tag{7.12}$$

$$r_{mn}^1 = \frac{1}{2}(1 - \delta r_{mn}) \tag{7.13}$$

where $\delta q_{mn} = (q_{mn}^0 - q_{mn}^1)$, and $\mathcal{N}(m) \backslash n$ represents the set of codeword bit nodes $\mathcal{N}(m)$ with node m removed; for example, if $\mathcal{N}(m)$ is a set of nodes {2,3,5,7}; then $\mathcal{N}(m) \backslash 3$ is the set of nodes {2,5,7}.

For each check node, the algorithm calculates the message it is about to send back to codeword node n by examining the received message from every other codeword node connected to it (that is, the expectation of the codeword nodes) and from them infers an expectation of codeword node n that satisfies the parity check equation that all connected codeword nodes must sum to zero. When all these messages are ready, they are sent back down the Tanner graph's edges to the codeword nodes.

Vertical step. The vertical phase of the algorithm is similar to the horizontal phase, but for each codeword bit. Having received the r-messages from the parity check nodes, each codeword node revises its estimate of its correct value. The algorithm proceeds to update the q-messages for each codeword node to send back to the parity check nodes, using the received r-messages as well as the prior probabilities, p, received from the demodulator;

$$q_{nm}^0 = \alpha p_n^0 \prod_{m' \in \mathcal{M}(n) \setminus m} r_{mn}^0 \tag{7.14}$$

$$q_{nm}^1 = \alpha p_n^1 \prod_{m' \in \mathcal{M}(n) \setminus m} r_{mn}^1 \tag{7.15}$$

$$\tag{7.16}$$

where α is a constant valued so that $q_{nm}^0 + q_{mn}^1 = 1$.

At this point *pseudo-posterior probabilities*, Q_m^0 and Q_m^1 for each codeword can also be found by inferring an expectation of the value of that node from all its information sources, so

$$Q_n^0 = \alpha p_n^0 \prod_{m' \in \mathcal{M}(n)} r_{mn}^0 \tag{7.17}$$

$$Q_n^1 = \alpha p_n^1 \prod_{m' \in \mathcal{M}(n)} r_{mn}^1 \tag{7.18}$$

$$\tag{7.19}$$

where α is a constant valued so that $Q_n^0 + Q_n^1 = 1$. From these probabilities, an estimate of $\mathbf{c}$ can be found.

$$\hat{\mathbf{c}}_n = \begin{cases} 1 & \text{if } Q_n^1 < Q_n^0 \\ 0 & \text{otherwise} \end{cases} \tag{7.20}$$

If $\hat{\mathbf{c}}$ satisfies the parity check equation $\mathbf{H}\hat{\mathbf{c}} = 0 \pmod 2$ then decoding is judged a success and the algorithm exits. If not then the algorithm returns to the horizontal

step, looping until it, hopefully, converges on a solution. There is generally a limit on the number of iterations performed in the algorithm, to force it to exit in a timely manner with a 'best guess' of the received codeword. Additionally, the algorithm can be programmed to exit when the differences in posterior probabilities through subsequent iterations become smaller than a defined minimum. Where received signals can be processed 'off-line', that is, time can be taken to decode the signals after the radio transmission has ended, more decoder iterations can be used to improve the noise performance of the decoder.

The result of decoding is the pseudo-posterior probabilities of the transmitted code word bits based on the prior probabilities received from the channel and the structure of the code. They are termed 'pseudo' because they are approximations of the result of the optimal decoder (an optimal decoder is possible to implement analytically in theory but is prohibitively computationally intensive in practice). These probabilities can be used to find the most likely codeword (equation 7.20). They can also be used with channel estimation and equalisation techniques such as turbo equalisation (see section 8.2).

The LDPC decoder is an intensive algorithm with computational complexity of order $6Nj$, where N is the block length and j is the LDPC matrix column weight [56].

7.5.5 Decoding in Log-Space

Repeated multiplications of very small floating point values in the LDPC decoder can result in a high incidence of *arithmetic underflow*, where the result of a calculation is two small to represent and is rounded to zero. This causes divide by zero errors later in the algorithm. This problem is exacerbated with larger code block lengths or good SNR conditions in the channel (due to the presence of $p(x = 0)/p(x = 1)$ terms in the algorithm, which tend to infinity as $p(x = 1)$ tends to zero). These errors must be corrected or otherwise compensated for in the algorithm, which normally adds an overhead. An alternative technique is to perform calculations in log-space, representing probabilities as log likelihood ratios (LLRs), which also provides a performance advantage as multiplies in linear space are equivalent to faster additions in log space. This technique was explored by Leung *et al.* [67][68] in the context of reducing the storage space and computational requirements of the LDPC decoding algorithm by representing probabilities as LLRs using a low number, N, of bits, in the form s^i where $i \in [0, \pm1, \pm2, ..., \pm(2^{m-1} - 1)]$.

Chung, Richardson and Urbanke describe a system of equations for performing LDPC decoding in log space [69]. All node values and messages are represented by their LLR equivalent, for codeword nodes these are

$$v_n = \text{LLR}(c_n) = \ln\left(\frac{p(x_n = 0|\mathbf{y})}{p(x_n = 1|\mathbf{y})}\right) \; 0 \leq n < N \tag{7.21}$$

where the parameters are the same as defined at the beginning of section 7.5.4. Messages from the parity check nodes are represented similarly

$$\mathbf{u_m n} = \ln\left(\frac{r_m n^0}{r_m n^1}\right). \tag{7.22}$$

Substituting equations 7.14 and 7.15 into 7.21, it can be readily shown that their log space equivalent is

$$v_{mn} = \text{LLR}(p_n) + \sum_{m' \in \mathcal{M}(n) \backslash m} u_{nm'} \tag{7.23}$$

where v_{mn} is the log space equivalent of q_{mn}, the message passed from codeword node n to parity check node m, and where $\text{LLR}(p_n)$ is the log-likelihood ratio of the n^{th} observed received bit from the demodulator. Finding the messages from the check nodes is less obvious and the derivation will not be described here, details can be found in [69]. They are found using the tan rule;

$$\tanh\left(\frac{u_{nm}}{2}\right) = \prod_{n' \in \mathcal{N}(m) \backslash n} \tanh\left(\frac{v_{mn'}}{2}\right). \tag{7.24}$$

When comparing the performance of normal and log-likelihood decoders it was found that the log likelihood decoder was approximately fifty percent faster than the linear one.

7.5.6 LDPC performance results

The bit error rates and frame error rates for four LDPC codes (three Mackay codes of codeword lengths between 204 to 816 bits and a QC-LDPC codeword of length 2044 bits), are presented in section 12.2 in Figures 12.2 and 12.3 respectively.

7.5.7 Application of LDPC to the communications link

The high performance of LDPC compared with other codes [56][3], as well as the innate asymmetry between encoder and decoder, make them ideal for use in the proposed modem (error rate performance for various codes are provided in section 12.2). The system specification from chapter 4 requires that the system have a high upstream bandwidth from the battery powered field unit to the mains powered base station, but only a simple downstream bandwidth for issuing commands. Using QC-LDPC coding, which allows a coder implementation of linear complexity on FSRs, will require minimal processing at the field unit, with the high complexity decoding process located at the base station. The downstream link need only use a simpler coding system.

Further to the high error correction performance of LDPC, there are powerful derived techniques that can also be incorporated into the project. The techniques examined that exploit the structure of LDPC codes are high performance channel equalisation (see section 8.2) and automatic repeat-request (see chapter 7.7.1).

7.6 Fountain codes

Fountain coding, alternatively known as the Luby technique or LT coding, is a technique based on sparse graphs and invented by Luby in 2002 [70]. These codes are designed for the erasure channel, the effect of which is to erase a given fraction of packets (no bit-flipping takes place). They have very high performance and are very simple to implement.

Fountain codes are the basis for a novel hybrid automatic repeat request technique described in section 7.10, which attempts to provide incremental redundancy to an LDPC encoded transmission, based on similarities between structures of the two techniques.

7.6.1 Generating fountain codes

To generate a fountain code, a message m_n, where $n = [1, 2, ..., N \times M]$, is split into N packets, $\mathbf{s}_n$, of M bits each. Each M-bit transmitted codeword $\mathbf{c}_i$ is the modulo-2 sum of a number of randomly chosen packets. The number of packets summed for codeword i is d_i and is chosen from a random distribution, $\rho(d)$. If we define $\mathcal{N}_i$ to be a set of d_i bits where $0 < x \in \mathcal{N}_i \leq N$ for all i and x then we can define the codeword

$$\mathbf{c}_i = \sum_{j \in \mathcal{N}_i} \mathbf{s}_j \pmod 2. \tag{7.25}$$

The shift registers used to generate the random numbers required for generation of d_i and $\mathcal{N}_i$ are required to be identical and synchronised between transmitter and receiver. Figure 7.5 gives a diagram of fountain coding on a bipartate graph. The message packets are lined at the bottom as message nodes, and the codeword nodes are at the top, and are valued as the modulo-2 sum of all connected message nodes. The codeword nodes are transmitted and the decoder has to infer the unknown message bits from a collection of received codewords.

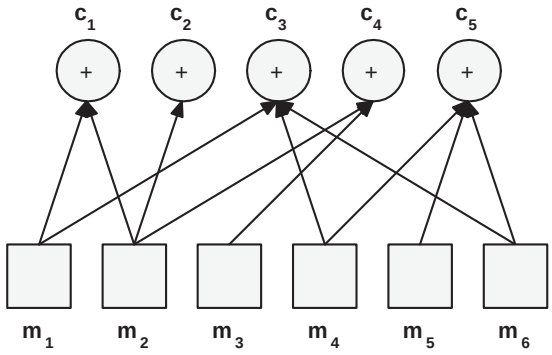


Figure 7.5: Fountain coding on a bipartite graph

7.6.2 Decoding fountain codes

To decode the original message, the receiver has to wait for receipt of N' packets, which is about 5% more than N for large packet sizes [70]. The overhead $N' - N$ is of order $\sqrt{N}(ln(N/\delta))^2$, where $1 - \delta$ is the probability of successfully receiving the packet [56]. On receipt of these messages, the receiver first searches for a received codeword of degree 1. If there are no codewords of degree one decoding

is declared a failure. If one such codeword is found (which will be denoted $\mathbf{t}_{i'}$), the corresponding data packet $\mathbf{s}_{n'}$ is set equal to it, where $n' = \mathcal{N}_{i'}$. Then any other codeword with $\mathbf{s}_{n'}$ in it (if $n' \in \mathcal{N}_i$) has its value added to it (in modulo-2) and its degree reduced by one, and $\mathbf{s}_{n'}$ is deleted. The algorithm then either terminates on the successful decode of all the source packets or starts again from the beginning, looking for more codewords of degree 1.

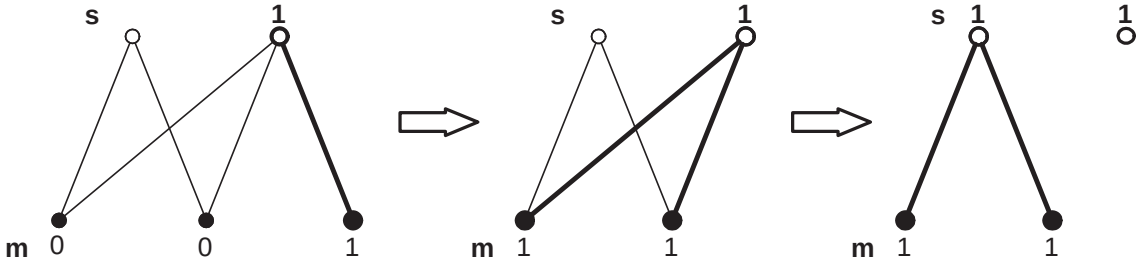


Figure 7.6: Fountain decoding on a bipartite graph

Figure 7.6 shows a very simple example of the fountain decoding technique on a bipartite graph. First a degree one codeword node is found and the corresponding source bit is set equal to it, the second step shows the other two connected codeword nodes having their values updated and the original selected bit is deleted. The third step shows the degrees of the updated nodes reduced by one (the edges connecting them to the resolved message node are effectively deleted), this leaves two codeword bits of degree one, the other message bit can be inferred from either of these two.

Fountain codes are known as *rateless* codes, in that they do not have any defined rate. The receiver needs to receive a certain number of codeword packets N' to decode the whole source message, as described above, but varying channel conditions may mean that more or less packets need to be sent through the erasure channel before the full data is received. It is clearly impossible to define a rate for these codes, although when transmitting fountain code, a maximum number of codeword packets might be defined for each message.

Fountain codes are very efficient on broadcast networks, such as video conferencing, where one server will transmit to many recipients. They provide a novel alternative to conventional automatic repeat request (ARQ) systems. When a message is broadcast to many users through an erasure channel then every user will ask for a resend of every packet that was erased on the route to them. It is unlikely that the same packets would be erased for all users so this leads to a large number of ARQs

which can saturate upstream and downstream bandwidth. Fountain codes provide a better solution in that the server may just keep sending out codewords until everyone has received the whole message. Each codeword sent will potentially help every recipient resolve their message, compared to conventional resends which will generally only benefit the requester.

7.6.3 Choice of distribution for fountain codes

The distribution, $\rho(d)$, that is used to choose the degree of fountain codes is an important metric that strongly affects the performance of the resulting codes. The set of N' fountain codes, which encodes N message bits, must satisfy two criteria to allow full decoding of the original message. First, the codes must completely define the message, i.e. every message bit must form part of at least one of the N' fountain codes, or it obviously won't be decoded. Second, there must be sufficient fountain codes of degree one to allow the decoder to finish decoding the data. In the best case only one node of degree one would be needed, but in practice, as the sets, $\mathcal{N}_i$ are generated randomly, the structure of the resultant graph may require more than one degree one node.

The ideal distribution, in expectation, is the *ideal soliton distribution* [56], which is expected to provide exactly one degree-one node and cover the rest of the nodes optimally. The distribution is given as

$$\rho(d) = \begin{cases} 1/N & \text{for } d = 1 \\ \dfrac{1}{d(d-1)} & \text{for } d > 1 \end{cases} \tag{7.26}$$

and the expected degree from this distribution is $\ln(N)$. This degree, however works poorly in practice on the erasure channel. With only one degree-one node expected for N codes, if it is erased then the decoding cannot commence and therefore fails. Also the propensity for the distribution to draw low degrees will in general cause low coverage of the original message with N codes. These two conditions entail that a large number of fountain codes would generally be required to decode the original data ($N' >> N$), which increases the overhead of the codes.

A better practical distribution, known as the *robust soliton distribution* improves the ideal soliton distribution by increasing the expectation of generating both degrees of one and degrees of a higher order that serves to increase the average coverage of

the codes. A distribution is defined as

$$\tau(d) = \begin{cases} \frac{S}{kD} & \text{for } d = 1, 2, \ldots (N/S) - 1 \\ \frac{S}{K}\ln(\frac{S}{\delta}) & \text{for } d = K/S \\ 0 & \text{for } d > K/S \end{cases} \tag{7.27}$$

where $S = c\ln(K/\delta)\sqrt{K}$ is the expected number of degree one checks and c and δ are parameters used to adjust the distribution. In practice if $N' = N + 2\ln(S/\delta)S$ are received the whole input message will be decoded with probability $1 - \delta$, [70]. The parameter c is left as a free parameter. Good results are observed when c is set smaller than 1 [56]. Equation 7.27 is added to $\rho(d)$ and normalised to obtain the robust soliton distribution

$$\mu(d) = \frac{\rho(d) + \tau(d)}{\sum_d (\rho(d) + \tau(d))} \tag{7.28}$$

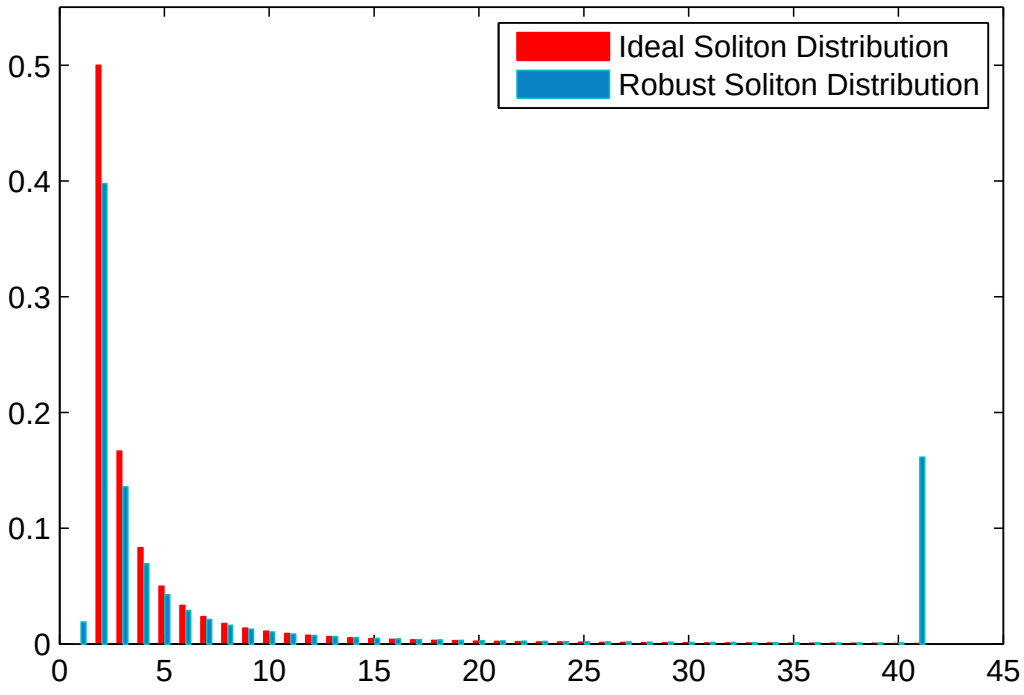


Figure 7.7: The soliton distributions

Figure 7.7 shows histograms of the ideal and robust soliton distributions with $N = 10000$, $c = 0.2$ and $\delta = 0.05$. The spike observed at the right of the robust soliton distribution is inherited from $\tau(d)$ and gives a high probability that every source

message bit will be included in the set of generated fountain codes. The robust soliton distribution can also be seen to have a higher probability of generating a one for the required degree one nodes.

In practice the overhead $(N' - N)/N$ required for full recovery of the source data will be about 5% for message lengths of $N \approx 10,000$ with well chosen δ and c [56].

7.7 Automatic repeat request

Automatic repeat request (ARQ) is another form of error correction, complementary to FEC channel coding in that errors are corrected by feeding back the presence of errors in the received packet from the receiver to the transmitter via some feedback path. It is an instructed error correction technique in contrast to the blind error detection and correction ethos of FEC.

Automatic repeat request brings with it a different set of challenges to FEC. Most importantly, it assumes the availability of a feedback channel capable of relaying information back from the receiver of a data signal to the transmitter. This may be problematic when running an ARQ system in an environment with a very poor channel, where it can be assumed that the feedback path will be no better than the main signal path. At the very least, an ARQ system requires some means of confirmation that a sent packet has been received correctly, this entails that the feedback path ought to be able to carry acknowledge (ACK) or negative-acknowledge (NAK) packets reliably, that when such a packet is sent, there is a very high chance of receiving it correctly at the remote unit. If these packets are difficult to relay, taking too long to feed back or having a high likelihood of failure, this will have a serious impact on the efficiency of an ARQ communication system.

As discussed above, an ARQ system required some means of confirmation that a packet has been received correctly. This necessitates that some form of error detection such as cyclic redundancy checking (CRC) be added to the data packets before transmission. This clearly adds an overhead to the system, reducing its efficiency. Furthermore, in a simple system which requests full resends of any packets not correctly received, further inefficiency is created; if one or two bits of a packet are received incorrectly this will cause an error to be detected and the entire packet to be resent, whereas a better solution would be to resend only those bits with likelihood

ratios closest to one (i.e. those with maximum uncertainty).

A well constructed ARQ system provides an adaptive level of redundancy to a communications system, in good conditions the redundancy is reduced to a minimal amount and as conditions deteriorate this redundancy is stepped up incrementally on a packet-by-packet basis. This is in contrast to FEC which generally uses a predetermined code rate, which may prove to embed more redundancy than is required for reliable communication over a given channel. The benefits of this adaptive redundancy are offset somewhat by the added complexity of implementing a two-way communications channel.

7.7.1 Hybrid ARQ

Hybrid automatic repeat request (HARQ) systems are, as the name suggests, techniques which combine FEC and ARQ techniques into a hybridized error correction system. Where FEC coding provides a baseline redundancy to a data signal, ARQ provides incremental increases to the signal's redundancy on an as-needed basis and therefore finds application in situations where guaranteed delivery is required and variable channel conditions are present. In the nomenclature, type I HARQ refers to a system which discards erroneously received data packets and resends the entire packet, often at a lower FEC rate. Type II HARQ keeps a buffer of the received vector and adds the information content of subsequent received repeats to it, in this case it is not necessary to retransmit all of the original message.

The simplest form of type II HARQ, chase combining [71], requires the receiver to average subsequent received repeats with the existing message vector. It is possible to exploit the structure of the FEC code and decoder to resend those parts of the packet that are most likely to add most to the information content of the received vector. Reliability-based HARQ (RB-HARQ) [72] is a technique whereby the receiver instructs the transmitter to resend those bits of the codeword with the lowest posterior likelihood magnitudes after decoding, but requires a high-bandwidth feedback link from receiver to transmitter which may be difficult in practice. Distribution-based HARQ (DB-HARQ) [73][74] is a technique used with irregular LDPC codes which randomly resends codeword bits with a bias in the favour of resending those bits which correspond to nodes in the Tanner graph of high degree (i.e. those with most connections).

The alternative form of type II HARQ is a protocol that resends extra encoded data (such as parity check bits) on each repeat, providing incremental redundancy to the received information vector. Approaches based on this technique have been developed to incrementally transmit vectors representative of punctured versions of the same codeword, which increases throughput adaptively when channel conditions are good [75] and sending supplementary parity check bits generated from a set of rate-compatible LDPC codes [76].

7.8 Fountain Codes and HARQ

Fountain codes, as described above, are sparse graph based codes that operate on the erasure channel. The effect of the erasure channel on a signal is that a given percentage of packets are erased completely and all the rest are received without error. The erasure channel is apt for simulating a channel with ARQ; each packet is assumed to carry a parity check such as CRC or similar which the receiver checks. If the parity check fails, the packet is treated as erased. If the check passes the packet is assumed received without error. The erasure channel is therefore, in this context, seen to exist as a logical layer on top of the combination of the physical channel (which could be AWGN or a fading channel) and the channel coding scheme, where an erasure is detected by the non-fulfilment of the coding scheme's parity check.

New fountain codes are generated to encode a packet of data and sent out from the transmitter constantly until the receiver acknowledges that the packet is received. There is no upper limit on how many fountain codes can be sent out for one packet, as each one is simply the linear combination of a subset of bits from the source packet. The absence of a code rate therefore leads to fountain codes being referred to as *rateless*. The subset of source bits to combine is determined at random, using a feedback shift register to generate the random numbers required which is shared by, and synchronised between, the transmitter and the receiver. A set of fountain codes is therefore treated, at the receiver, as a system of linear equations. If the set describes every bit in the encoded packet then the packet can be decoded (without error due to the effects of the erasure channel) and the receiver sends an ACK signal in reply, which instructs the transmitter to send out fountain codes for the next packet. The random number generators are kept in sync implicitly as the sender and receiver both keep track of how many messages have been sent and resent.

7.8.1 Similarities between fountain codes and LDPC

The erasure channel, as described in the previous section, sits as a logical layer on top of the channel coder, and a parity check on the decoder output is required to inform the receiver of whether data was received correctly or incorrectly. LDPC has an inbuilt parity check (indeed LDPC is decoded by finding the most likely input which satisfies this check, see section 7.5.4), so an LDPC decoder can be assumed to output correct data if the parity check is fulfilled, or incorrect data if not, after a given number of decoder iterations. Thus, the path of data being fed into the LDPC encoder, sent across a channel and decoded is representative of an erasure channel, so long as decoded data that fails its parity check is marked erased.

A systematic LDPC codeword consists of the original message vector appended with a number of parity check bits such that the generated codeword fulfils the parity checks described by the parity check matrix. Let $\mathbf{H}$ be defined as an $N \times M$ parity check matrix for an LDPC code. Let $\mathbf{w}_j$ represent the rows of $\mathbf{H}$, where $0 < j \leq M$. Let $\mathbf{c}$ be a valid codeword for $\mathbf{H}$. The parity check condition is given as

$$\mathbf{Hc} = \sum_j \mathbf{w}_j.\mathbf{c} = \mathbf{0} \pmod 2 \tag{7.29}$$

where $\mathbf{0}$ is a zero vector and the middle equality is the expansion of the matrix multiplication with respect to the rows of the $\mathbf{H}$. By examining a single row and further expanding

$$\mathbf{w}_j.\mathbf{c} = \sum_i w_{j,i} c_i = \mathbf{0} \tag{7.30}$$

is found where c_i is the ith element of $\mathbf{c}$ ($0 < i \leq N$), and $w_{j,i}$ is the ith element of $\mathbf{H}_j$. Because $\mathbf{w}_j$ is composed of only zeros and ones, it is possible to replace the $\sum_i w_{j,i}$ part of equation 7.30 with a sum over a set, $\mathcal{N}_j$ of indices given by the locations of the ones in $\mathbf{w}_j$

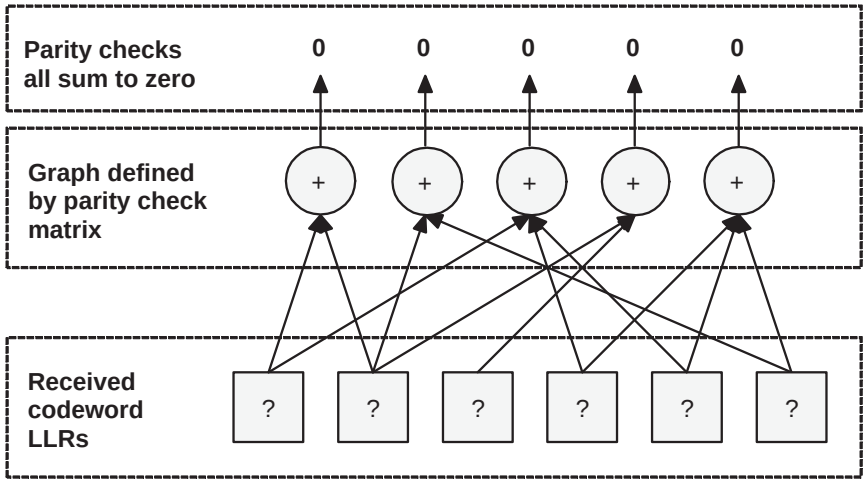
$$\sum_{i \in \mathcal{N}_j} c_i = 0 \tag{7.31}$$

which is seen to be very similar to equation 7.25 for fountain code lengths of one. That equation is reproduced here with some minor alterations

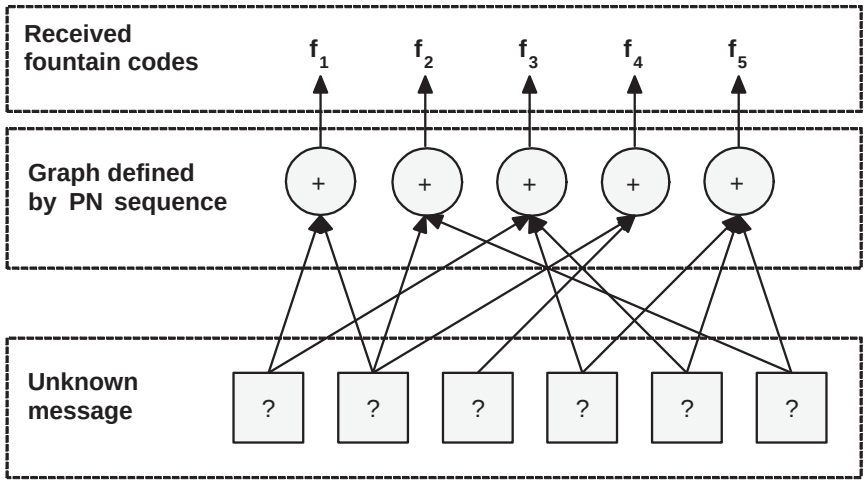
$$f_k = \sum_{i \in \mathcal{M}_k} c_i \pmod 2 \tag{7.32}$$

where f_k is one of the k one-bit fountain codes, generated by summing together bits, c_i, of the source packet. $\mathcal{M}_k$ is the kth set of indices. A fountain code is thus

shown to be very similar to an LDPC parity check (which each row in **H** represents), the only difference being the sum of the codeword bits. The sum of the component codeword bits of an LDPC parity check are assumed to equal zero for decoding. However the sum of the component bits of a fountain code are assumed to sum to the value of the fountain code, f_k.



(a) LDPC



(b) Fountain codes

Figure 7.8: Diagrams showing the structural similarities of LDPC and fountain codes

Figure 7.8 provides a comparison of the structure of LDPC codes and fountain codes. The received probabilities of the LDPC code are fed into the bottom box in Figure 7.8(a) and the decoder finds the most probable codeword represented by these probabilities given that all the parity checks sum to zero. In Figure 7.8(b) received fountain codes are fed into the top box, and the decoder infers the original message from these.

Soft decoding of fountain codes using a similar belief-propagation algorithm, as used in LDPC decoding, for wireless communications over any channel has been investigated by Jenkac et. al. and proven to be a well performing method [77].

7.9 HARQ using LDPC and fountain coding

LDPC codes are known to perform well in noisy and fading channels, with throughput arbitrarily close to the Shannon limit when using long block lengths [39][56]. That, in combination with its intrinsic parity check makes it a good choice for the FEC technique in a HARQ system. Fountain codes are known to perform very well for correcting erasures and the mechanism for providing this performance is connected to the properties of the sparse graph upon which it is based, similarly to LDPC. The similarity between fountain codes and LDPC codes makes it possible for fountain codes to act as an 'addendum' to an LDPC code, which acts to provide further information to the belief propagation algorithm, increasing its performance and assisting it on converging to an error free result.

LDPC and fountain codes can, as discussed above and in Figure 7.8, be represented as Tanner graphs. When receiving an LDPC codeword sent through a noisy channel, the prior probabilities of each codeword bit are placed in the bottom row of cells of the graph. The number of cells in the top row is determined by the number of rows, or parity checks, in the parity check matrix. The top row and bottom row of cells in an LDPC Tanner graph are referred to as the parity check nodes and codeword nodes respectfully. The links between the parity check and codeword nodes are determined by the position of ones in the parity check matrix, so if the first row in the matrix had ones at locations 3, 5 and 15, the first parity check node has connections to the first, third and 15th codeword nodes. Figure 7.8(a) shows this. Fountain codes are similar; to encode, an input message is placed in the bottom row of cells (message nodes) and a pseudo-random sequence, used to generate sets of indices, describes the links between the top row of cells (sum nodes) and the bottom row. The transmitted code in this case is the output of the sum nodes. Each one represents a one-bit fountain code, see Figure 7.8(b).

If an LDPC codeword generated from an $N \times M$ parity check matrix were used to encode K one-bit fountain codes, then the Tanner graphs for each technique could be merged together by using a common bottom row. This would result in a Tanner

graph with N nodes in the bottom row and $M + k$ nodes in the top row. Figure 7.9 illustrates this situation. The process of merging the graphs effectively adds top row nodes, like extra parity check nodes. It is important to remember that these extra nodes are not technically parity check nodes as they do not sum to zero, they would more correctly be referred to as sum nodes or fountain nodes.

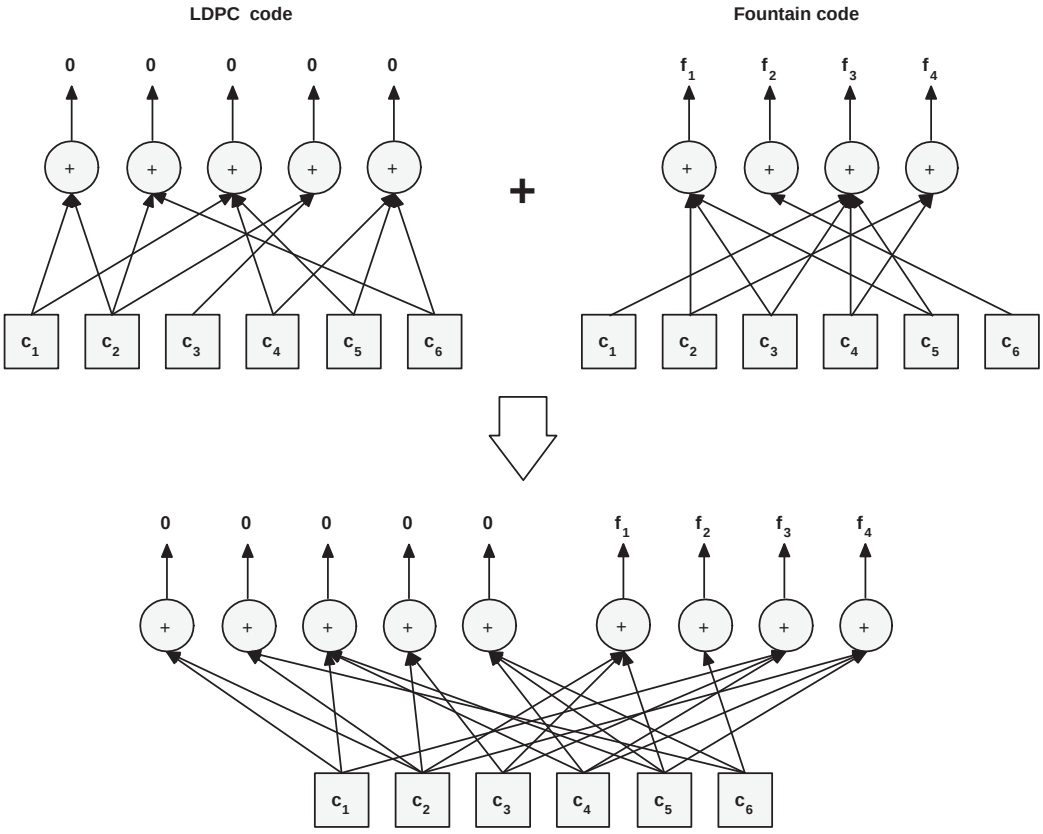


Figure 7.9: Diagram showing how the Tanner graphs of LDPC and fountain coding can be combined

7.10 Proposed new HARQ system

The fundamental difference between chase combining and RB-HARQ is that RB-HARQ instructs the transmitter to resend those bits that it is most unsure about to the receiver, thus updating only those received bits most corrupted by noise. Unfortunately the high bandwidth feedback link required from the receiver to the transmitter makes RB-HARQ difficult or impossible to implement in a real world situation.

Performance results for RB-HARQ are given in section 12.5. The performance improvement demonstrated by using RB-HARQ over chase combining proves that

resending information pertinent to the lowest likelihood bits improves throughput performance over random resends. The problem is that the distribution of posterior likelihoods at the receiver is unknown by the transmitter which, in the case of RB-HARQ requires a high bandwidth feedback link to relay back this information. Therefore, for a technique to be successful, the likelihood of resending information corresponding to low posterior likelihood bits at the receiver must be maximised. From this premise, a novel repeat request technique, based on fountain codes as described above, was developed that combines random sets of codeword bits from the transmitter into packets sent as repeat information to the receiver.

Sending a random combination of bits increases the likelihood of sending information pertinent to the lowest likelihood bits as it increases the number of bits which participate in each repeat. It is proposed that sending the modulo-2 sum of a subset of codeword bits will improve performance beyond that of chase combining, providing performance closer to that of RB-HARQ without requiring a high bandwidth feedback path. It is assumed that the transmitter and receiver contain random number generators synchronised to each other from which it is possible to generate sets of n indices to the codeword bits. Let these sets of indices be denoted $\mathcal{N}_f(k)$ where $0 < k \leq K$ is the set index. If K bits in total are resent, then each resend is calculated as

$$f_k = \sum_{n \in \mathcal{N}_f(k)} c_n \,(\text{mod } 2). \tag{7.33}$$

For clarity, this technique will be referred to as 'random sum of elements HARQ', or RSE-HARQ from here on.

It is useful to consider the similarities between a parity check and an RSE-HARQ repeat: a parity check is equivalent to the special case of an RSE-HARQ repeat where $f_k = 0$ which is received without noise (that is with a prior probability $P(f_k = 0|y_k) = 1$ where $y_k = f_k + \eta_k$ is the received symbol with noise η_k). This equivalence gives insight into modifications to the belief propagation algorithm necessary to decode the repeat bits. These modifications are described below.

7.10.1 Modifications to the belief propagation decoder

The belief propagation algorithm is the most popular technique for decoding LDPC. As described above it is a message passing algorithm and is split into two phases; the vertical phase updates the messages sent from the codeword nodes on the Tan-

ner graph of the parity check matrix to the parity check nodes and the horizontal phase computes the messages sent from the parity check nodes to the codeword nodes and updates the estimate of posterior probabilities. The two steps are iteratively executed with messages being passed from the codeword nodes to the parity check nodes and back until the codeword nodes converge on a pattern that satisfies the parity check equation, at which point the algorithm declares decoding a success. If the parity check equation is not satisfied after a given number of iterations then the algorithm stops and declares decoding unsuccessful. However the resulting pseudo-posterior probabilities from the decoder are likely to be a better estimate of the original codeword than the prior probabilities.

As discussed above, there is an equivalence between a parity check node and a RSE-HARQ repeat whereby they are the same for the special case $P(f_k = 0|y_k) = 1$. This property is embedded in the parity check matrix as prior knowledge of the structure of the code. The RSE-HARQ repeat bit differs in that there is no prior knowledge of the summation of the codeword bits in the set $\mathcal{N}_f(k)$, and this information is provided by the received repeat bit probability $P(y_k)$. From this, each RSE-HARQ repeat bit can be represented by an 'extra' parity check node, modified to take into consideration the received value of y_k.

As adding extra RSE-HARQ repeat packets to the end of an LDPC code is fundamentally the same as adding extra 'parity check' nodes incrementally to the sparse graph, similar to how sending fountain codes provides incremental increases in complexity to the structure of the sparse graph. Therefore to use these extra bits to decode the original LDPC code, each RSE-HARQ node will be processed in turn as part of the horizontal phase of the BP algorithm. Similarly to the LDPC parity check nodes a most likely return value for every connected codeword node will be computed, taking into consideration the incoming messages from all other connected sources. In LDPC these sources are the messages from each connected codeword and the knowledge that the sum of each connected codeword must be zero (as it is a parity check). For LDPC/RSE-HARQ, the messages from each connected codeword nodes are again sources, but now instead of assuming that their sum is zero, the decoder takes into consideration the prior probability of the corresponding received fountain code, and assumes the value of all the connected codeword nodes is equal to it. In the vertical phase of the decoding algorithm, the decoder works similarly to an LDPC decoder. In an LDPC decoder each codeword node is taken in turn and sends a message to each parity check node based on the messages from every other connected node and the prior probability of the corresponding received

codeword bit. For the LDPC/RSE-HARQ decoder the fountain nodes are treated just as any other parity check node, so each codeword node sends a message to each connected node (both parity check and fountain types), based on the messages from every other connected node (both types).

Horizontal step The equation for the horizontal step of the BP algorithm is given here

$$\delta r_{mn} = \prod_{n' \in \mathcal{N}(m) \setminus n} \delta q_{mn'} \tag{7.34}$$

where r_{mn} is the message to be passed from parity check node m to codeword node n, $\delta r_{mn} = (r^0_{mn} - r^1_{mn})$, and q_{mn} is the message passed from codeword node n to parity check node m, $\delta q_{mn} = (q^0_{mn} - q^1_{mn})$. $\mathcal{N}(m)$ refers to the set of indices given by the placement of ones in row m of the parity check matrix. This equation is performed for every edge on the Tanner graph to complete the horizontal phase.

For an LDPC codeword generated from an $N \times M$ parity check matrix with K supplementary one-bit RSE-HARQ repeats sent with it, there are M parity check nodes and K RSE-HARQ nodes, giving $P = M + K$ top row nodes in total. These nodes are numbered $0 < m \leq P$, where the nodes with $i \leq M$ are parity check nodes and the nodes with $M < i \leq P$ are fountain nodes. A pseudo-random process, shared between transmitter and receiver, is used to generate the sets on indices required for each code; these sets are denoted $\mathcal{N}_f(k)$, where $0 < k \leq K$. The first M nodes are no different than in a simple LDPC code, so these nodes use equation 7.34 for $0 < m \leq M$ to update messages to their connected nodes. However the last K nodes must be treated differently to account for the different information source, i.e. the existence of a received prior probability, for the sum of all connected codeword nodes, instead of prior knowledge that the sum should be zero. The prior probabilities received are represented by f^x_k; f^0_k represents the probability that f_k is zero, $P(f_k = 0)$, and f^1_k represents $P(f_k = 1)$. This incoming probability need not be treated any different to the other probabilities, as it is just another information source. So equation 7.34 can be modified to accommodate it easily

$$\delta r_{mn} = \delta f_{m-M} \prod_{n' \in \mathcal{N}_f(m-M) \setminus n} \delta q_{mn'} \tag{7.35}$$

for $M < m \leq P$, where $\delta f_m = (f^0_m - f^1_m)$. Combining this with equation 7.35, the

complete horizontal step equation is

$$\delta r_{mn} = \begin{cases} \prod_{n' \in \mathcal{N}(m) \setminus n} \delta q_{mn'} & 0 < m \leq M \\ \delta f_{m-M} \prod_{n' \in \mathcal{N}_f(m-M) \setminus n} \delta q_{mn'} & M < m \leq P \end{cases} \tag{7.36}$$

Vertical step Using the example above of an LDPC code generated from an $N \times M$ parity check matrix supplemented by K one-bit RSE-HARQ packets, there are no additional codeword nodes added by the received RSE-HARQ bits. These codes are generated from the codeword output by the LDPC encoding, hence the Tanner graphs of the LDPC code and the fountain codes are combined with a common bottom row as in Figure 7.9.

The equations for the vertical step of the LDPC BP algorithm are reproduced here

$$q^0_{nm} = \alpha p^0_n \prod_{m' \in \mathcal{M}(n) \setminus m} r^0_{mn} \tag{7.37}$$

$$q^1_{nm} = \alpha p^1_n \prod_{m' \in \mathcal{M}(n) \setminus m} r^1_{mn}. \tag{7.38}$$

$$\tag{7.39}$$

These equations update the messages, q^x_{nm} passed from the codeword nodes back to the parity check nodes, where x is zero or one. $\mathcal{M}_n$ denotes the set of parity check nodes connected to codeword node n (or equivalently the indices of parity checks that codeword bit n participates in). The prior probabilities of the codeword bits are denoted p^x_n. The vertical step goes through every codeword node and evaluates the messages to be sent to every connected parity check node based on the product of the messages received from every other node, as well as the received prior probability of the corresponding bit from the demodulator. Posterior probabilities for the codeword are generated in a similar way (see section 7.5.4 for details) which are used to generate an estimate for the codeword. If this estimate fulfils all the parity checks the coding is deemed a success and the algorithm exits. On the first iteration of the algorithm, the q messages are set simply to the received prior probabilities of their connected codeword nodes.

Extending this step of the algorithm to support the extra fountain nodes is relatively simple. As the fountain code nodes are treated the same as an ordinary parity check node, the algorithm simply has to incorporate the set of fountain nodes connected to each codeword node in the equation. Define $\mathcal{M}_f$ to be the set of RSE-HARQ nodes connected to codeword node n, then we define a set $\mathcal{M}_a(n) = \mathcal{M}(n) \cup \mathcal{M}_f(n)$. The

equations for the vertical step of the BP algorithm then become

$$q_{nm}^0 = \alpha p_n^0 \prod_{m' \in \mathcal{M}_a(n) \setminus m} r_{mn}^0 \tag{7.40}$$

$$q_{nm}^1 = \alpha p_n^1 \prod_{m' \in \mathcal{M}_a(n) \setminus m} r_{mn}^1 \tag{7.41}$$

with the pseudo-posterior probabilities evaluated similarly.

An estimate of the received codeword can be evaluated from the pseudo-posterior probabilities. If this estimate is found to satisfy the parity check constraints then the decoding is judged a success and it is the receivers' responsibility to send an acknowledge signal back to the transmitter instructing it to move onto the next message. If after a set number of iterations the parity check equation is still not satisfied, then no acknowledge signal is sent and the transmitter continues to send further repeats until the receiver can successfully decode them.

7.10.2 Effects of varying random set size

A fundamental choice to be considered when implementing an RSE-HARQ system is deciding the degree of each set, which is the number of indices within each set $\mathcal{N}_f(k)$, denoted by n. For example an RSE-HARQ system with $n = 2$ would combine two codeword bits to form each repeat.

There are advantages and disadvantages to increasing n. It is advantageous as it increases the likelihood that a bit with low posterior probability at the decoder will be embedded within any given repeat. However decoding performance depends on there being only one bit within each set with a low posterior probability. If there are more than one then the additional degrees of freedom seen at the decoder are likely to inhibit performance. The probability of transmitting the sum of such 'good' sets of a given degree is linked to the signal to noise ratio of the received signal.

7.11 HARQ performance results

The relative throughputs of random repeats, reliability based HARQ and the new RSE-HARQ technique are shown in Figure 12.14 in section 12.5. Section 12.5.1

provides an evaluation of the effects of varying the set size for RSE-HARQ and the throughput results are presented in Figure 12.15.

Chapter 8

Channel equalisation

A communications channel will affect and degrade a signal in many ways, including adding noise, multipath, and Doppler shift to a signal passing through it. Fading, which causes random variations in the phase and amplitude of the received signal (where a *fade* is a point in time in which the received signal loses the majority of its power) is a consequence of both multipath and distortions, due to the signal passing through the atmosphere. When the signal is received these distortions must be compensated for if the data is to be correctly demodulated. *Channel equalisation* refers to the act of compensating for these distortions.

This chapter discusses the subject of channel equalisation and briefly describes some of the more well known equalisation techniques. It also details the particular technique chosen for use with the proposed communications system, blind LDPC turbo equalisation.

8.1 Equalisation of digital signals

A signal, when received from a communications channel, is subject to distortions and noise arising from the physical nature of the channel. These distortions can be represented, in the discrete domain, by a channel transfer function of length L, $\mathbf{h} = h_0, h_1, \ldots, h_{L-1}$. Under these conditions the received signal, $\mathbf{y}$, is the convolution of the transmitted signal, $\mathbf{x}$, and the channel impulse response. That is, $\mathbf{y} = \mathbf{x} * \mathbf{h} + \nu$, where ν represents noise added to the signal from the channel. Figure 8.1 shows a block representation of this paradigm for the case $L = 3$. To aid in the successful

decoding of data transferred upon the received signal, it is normally desirable that the effects of the channel are removed. To perform this operation the receiver must find an estimate of the channel impulse response, $\hat{\mathbf{h}}[t]$, and perform deconvolution of the received signal and the estimate to find an estimate of $\mathbf{x}[t]$. Further discourse on the basics of equalisation can be found in [3].

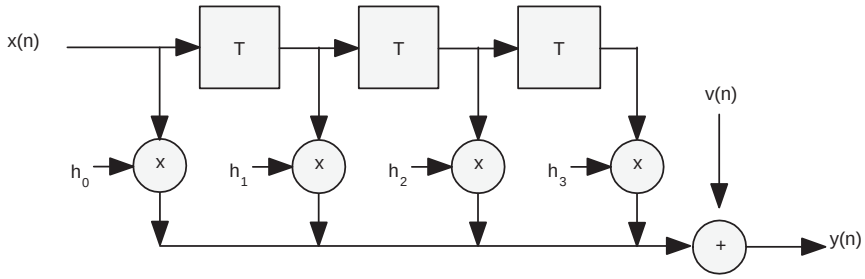


Figure 8.1: FIR representation of a channel

Time-variant fading channels such as those found in long distance HF communications provide a challenge in that the channel transfer function is changing continuously, resulting in random variations in amplitude and phase of the signal at the receiver. In some cases, such as for long packet lengths, the channel transfer function might change considerably over the duration of a single transmission. This often requires the use of a sophisticated equaliser at the receiver which can track changes in amplitude, as well as an interleaver or similar to combat the effects of deep fades where the received signal to noise ratio (SNR) of the received signal drops below that required for reliable decoding.

Generally, a training sequence is required to synchronise the transmitter and receiver, as well as provide some initial information on the channel to the receiver. The training sequence (otherwise known as the *channel probe* sequence) is transmitted with the data payload. As the training sequence is also known at the receiver, an estimate of the channel can be found by correlating the known sequence with the received signal.

Blind equalisation is a technique which attempts to find the channel parameters from examining the metrics of the received signal and matching them with the known metrics of the signal [78][79]. This generally requires higher order signal parameters to be found from the incoming signal, which can be a computationally intensive process.

8.1.1 Single tone equalisation

Single-tone equalisation of signals, especially in poor channels such as those seen in the HF channel, is a challenging task. For common single tone RF modulation techniques such as phase shift keying (PSK) and quadrature amplitude modulation (QAM), which have a constellation of symbols whereby the phase and amplitude of a carrier wave are manipulated to represent bits of data, the corruption resulting from transmitting such a signal through a poor, fading channel can cause large error rates at the receiver.

An intuitive way of correcting phase and amplitude distortions in the digital domain is to use a finite impulse response (FIR) filter. As the carrier frequency of the signal generally remains constant, it is possible to design a filter that can perform phase and amplitude corrections so long as they remain constant. In most real channels, however, the phase and amplitude distortions are continuously varying and the taps of any such correcting filter must therefore be updated regularly. Channel distortions can be measured by correlating the output of the received signal with a probe sequence, which is a sequence of high spectral content known to both the transmitter and receiver, to give a good initial estimate of the transformation caused by the channel by comparing the received version with the known version. An adaptive algorithm can be used which accepts the output of the equaliser FIR filter as input and adjusts the tap coefficients to make the output of the filter equal to the known probe data [80]. After the probe has been sent to initialise the state of the equaliser, data can be sent, and the adaptive algorithm in the equaliser will attempt to keep the constellation of the received data correct, by minimising the mean squared error of the equalised data compared to the constellation points. During deep fades or momentarily poor channel conditions, say with strong impulsive noise, it is possible that the equaliser might lose track of the correct channel transform and so for this reason short probe sequences are periodically sent with the data to help the equaliser correct for such errors. The MIL-STD-118-110B waveform [20], which is a common format for data transmission in HF communications, sends probe data in this way. This type of equaliser is known as a *transversal* or *linear* equaliser.

Another common equaliser, which is a modified version of the equaliser described above, is the decision feedback equaliser (DFE) [10][3]. This technique builds upon the transversal equaliser by subtracting the effects of the last detected symbols from the symbol currently being detected. This serves to combat inter-symbol interference resulting from the delayed components of previous symbols in the presence

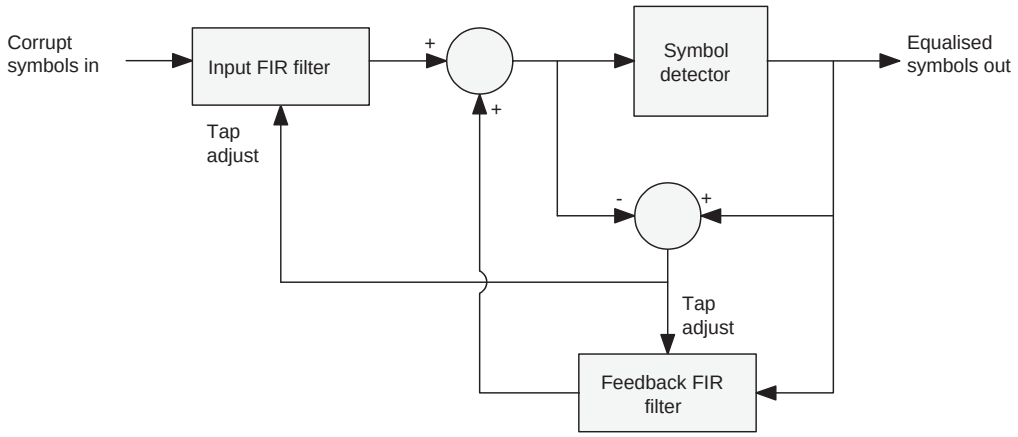


Figure 8.2: Block diagram of a decision feedback equaliser (adapted from [3])

of a channel with spectral nulls. Figure 8.2 shows a block diagram of the DFE, it is similar to the transversal equaliser but also implements a feedback filter to cancel the effects of the previous symbols. At the output a symbol detector is used which outputs the closest symbol in the constellation to the corrupt symbol which is input. The equaliser attempts to make the input and output of the symbol detector equal, at which point the system is operating identically to the transversal equaliser described above.

For further reading on single tone equalisation techniques and adaptive filter algorithms please refer to [10][80].

8.2 Turbo Equalisation

Turbo equalisation [81] is an iterative method that uses information from an FEC decoder to help equalise the channel output, it is an adaptation of the turbo decoding algorithm and first saw use equalising received data coded with this technique. Turbo codes are generated by concatenating the results of the outputs of two convolutional coders, one which takes the original data as its input (output one), and one that takes an interleaved version of the data as its input (output two) [56]. To decode the resulting signal two *maximum a posteriori* (MAP) decoders are used, one each for the (deconcatenated) outputs of the encoder. The first one takes output one from the encoder, performs a MAP decoding on it to find the most likely path through the convolutional code's trellis and outputs the bit likelihoods of the input sequence (similar to the Viterbi algorithm). These have the *a priori* (AP) probabilities of the

decoder (which are extrisic likelihoods, $P_{2,ext}$, from the second decoder and are all initialised to 0.5 for the first iteration) subtracted from them, forming the extrinsic posterior likelihoods, $P_{1,ext}$. These are then interleaved using a matching interleaver to the one in the encoder, and the output of this forms *a posteriori* probabilities (APP) for the second MAP decoder. The second decoder receives the output of the convolutional coder that works on the interleaved data, uses this and $P_{1,ext}$ from the first decoder to find the APP for the output of the second decoder. This is then fed into a deinterleaver which performs the inverse of the encoder's interleaver. The result is the APP of the turbo code itself, that is, the probabilities of the string of bits originally used to generate the turbo code. The likelihoods from the first decoder, $P_{1,ext}$, are subtracted from these values, to form the extrinsic likelihoods $P_{2,ext}$, and these are fed back as the prior probabilities of the first decoder. Extrinsic likelihoods are used to ensure that the decoders are only working on new information, and prevent a positive feedback loop from forming.

The Bahl, Cocke, Jelenik and Raviv (BCJR) algorithm [82] is used as the maximum a posteriori decoder in the turbo decoding algorithm. This technique was first introduced in the 1970s, but did not see widespread use for a long time as it was more complicated to implement than the Viterbi algorithm, and used for the same applications. The Viterbi algorithm is not suitable for turbo equalisation because it outputs hard decisions; the decoder eliminates paths that it deems unlikely. The elimination of these paths causes a loss of information that is necessary for turbo equalisation to effectively take place. The BCJR algorithm, however, finds probabilities for every possible path through the trellis and outputs the total probability for each bit. This algorithm is further discussed in section 8.2.1 below.

Turbo equalisation is an adaptation of the turbo decoding algorithm that equalises the channel in addition to decoding the FEC code. It can as such only be used with maximum likelihood iterated decoding schemes such as turbo or LDPC decoding. From this point onwards, descriptions of turbo decoding will focus on its implementation with an LDPC decoder, as this is the forward error correction scheme chosen to be used in the final system.

The turbo equalisation algorithm first assumes that the channel can be modelled as an length L FIR filter, with coefficients $\mathbf{h} = [h_0, h_1, ..., h_L]$ and white Gaussian noise $\nu(n)$ added at the output. This is represented in Figure 8.1 for the case $L = 3$. A source message is encoded into an LDPC codeword, $\mathbf{c} = [c_1, c_2, ..., c_N]$ of length N is modulated into the symbols $\mathbf{x} \in \mathcal{A}$ where $\mathcal{A}$ is the constellation of J complex symbols

used in the modulation. This forms the input to the channel. There are J^L states that the channel can be in at any one time, corresponding to every combination of modulated symbols in the channel model's FIR registers. The observations from the channel are given by $\mathbf{y} = [y_1, y_2, ..., y_{L+N}] = \mathbf{h} * \mathbf{x}$. Given a channel estimate, $\hat{\mathbf{h}}$, the object of the turbo equalisation algorithm is to infer the posterior probabilities of received codeword given the received data and knowledge of the set of possible modulated symbols and the code structure, $P(\mathbf{c}|\mathbf{y}, \mathcal{A}, \mathbf{A})$ where **A** is the LDPC parity check matrix.

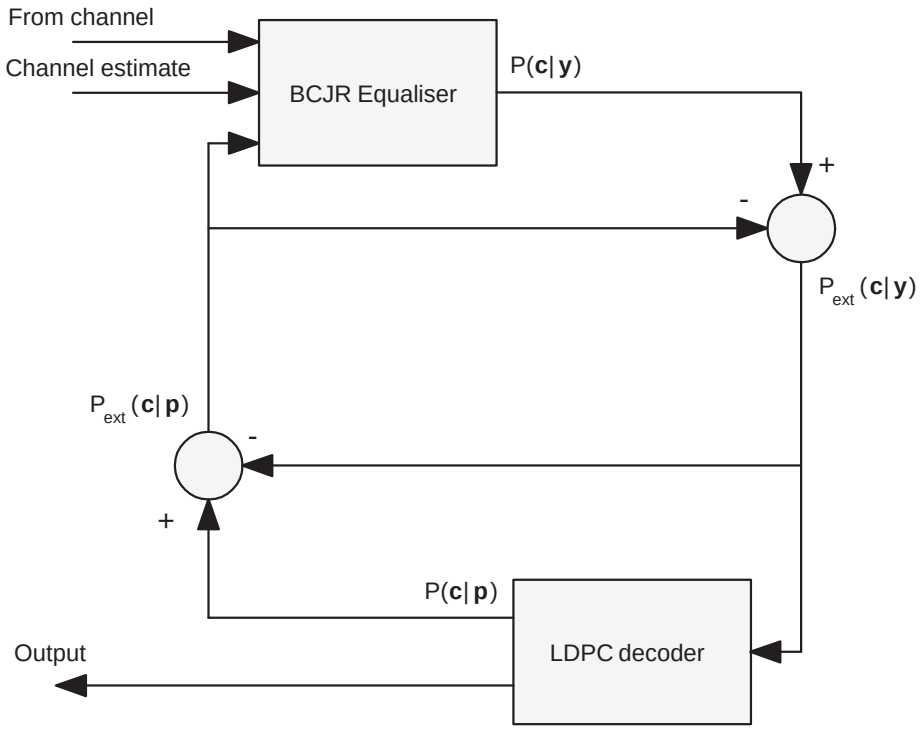


Figure 8.3: Block diagram of an LDPC turbo equaliser

Figure 8.3 presents a block diagram of a basic turbo equaliser. The received symbols are fed into the BCJR equaliser, which finds the probabilities of the codeword bits used to generate them, $P(\mathbf{c}|\mathbf{y}, \mathbf{h})$ assuming that an accurate channel estimate is also provided. These probabilities have the extrinsic codeword probabilities from the LDPC decoder subtracted from them, giving the extrinsic probabilities $P_{ext}(\mathbf{c}|\mathbf{y}, \mathbf{h})$. The extrinsic probabilities are used as prior information for the LDPC decoder which outputs posterior probabilities for the codeword bits $P(\mathbf{c}|\mathbf{p})$, where $\mathbf{p} = (P(c_1|\mathbf{y}, \mathbf{h}), P(c_2|\mathbf{y}, \mathbf{h}), ..., P(c_N|\mathbf{y}, \mathbf{h}))$. The extrinsic information which was fed into the LDPC decoder is subtracted from these probabilities (similar to the previous extrinsic probability calculation) forming the extrinsic probabilities from the decoder output, $P_{ext}(\mathbf{c}|\mathbf{p})$. These are then used to determine state transition probabilities in the BCJR equaliser for the next iteration, improving its estimate. Each iteration suc-

cessively improves the estimate from the BCJR algorithm, equalising the channel, and decoding finishes after a set number of iterations or when the LDPC decoder successfully decodes the signal (such that $\mathbf{A}\hat{\mathbf{c}} = 0 \pmod 2$).

8.2.1 BJCR equaliser

The BJCR equaliser [82][81] performs demapping of the received symbols. It converts the incoming stream of corrupt symbols into posterior probabilities of the codeword bits used to generate them by treating the channel as a hidden Markov model, with one state for each of the J^L possible states of the channel model, and state transition probabilities derived from extrinsic probabilities from the output of the LDPC decoder. The BCJR algorithm is a trellis based maximum a-posteriori (MAP) decoder. Otherwise known as the forward/backward algorithm, it performs passes forwards and backwards through the trellis, generating overall probabilities for each state at each stage in the trellis.

The BCJR equaliser, is a maximum likelihood sequence estimation (MLSE) technique which treats the channel as a finite impulse response (FIR) filter of length L and n_p complex taps coming of some or all of the elements of the shift register, which correspond to the magnitude and phase distortions of each of the n_p multipath components of the received signal. The equaliser implements a trellis with 2^{L-1} possible states, which represents possible states of the channel FIR. The state transitions between the nodes of the trellis represent the changes to the channel's shift register stemming from a new symbol entering the channel. Each node has N state transitions exiting it where N is the size of the symbol set used in for modulation (for example, with BPSK, $N = 2$).

The algorithm takes as input the received symbols, state transition probabilities, an estimate of the noise variance and an estimate of the channel and first finds the probabilities of each state transition at each point in the trellis, $\gamma_{i,j}^{s_x}$, where i is the previous state, j is the next state and s_x is the x-th symbol of the modulating constellation. It then proceeds to move up and down the received block of symbols, evaluating α and β which are the cumulative probabilities of each state looking forwards through time and backwards through time respectively. Finally it uses α and β to find the posterior probabilities of the sequence of symbols fed into the channel to generate the sequence of received symbols.

The algorithm effectively converts the likelihoods of the received symbols, $P(y_n|\mathbf{x}, \hat{\mathbf{h}})$ into the posterior probabilities $P(\mathbf{x} = a|y_1, y_2, ..., y_N, \hat{\mathbf{h}})$ for all $0 < n \leq N$ and $a \in \mathcal{A}$. From these likelihoods it is trivial to find posterior probabilities of the codeword bits $P(c_n|\mathbf{y})$. These probabilities have the outputs from the LDPC decoder subtracted from them to give the extrinsic probabilities $P_{ext}(c_n|\mathbf{y})$ which are used as prior probabilities for the BCJR equaliser.

For the first iteration of the decoder the channel state transition probabilities are assumed to be 0.5 for every stage in the trellis. This represents random transition probabilities, which are assumed on the first pass as the LDPC decoder at that point has no information. The BCJR algorithm in this first iteration therefore gives probabilities for the received symbols based solely on those symbols and the channel estimate. Further iterations cause additional information to be passed to the equaliser which allow it to refine its output.

8.3 Blind Turbo Equalisation

Blind turbo equalisation builds on the technique described above by incorporating a channel estimator to generate a new estimate $\hat{\mathbf{h}}$ in every iteration. The estimate is fed into the equaliser and the complete system iteratively minimises the difference between the estimated and actual channel coefficients, $|\hat{\mathbf{h}} - \mathbf{h}|$, thus decoding the source data. Figure 8.4 shows a block diagram of a blind turbo equaliser, it is identical to the standard turbo equaliser except there is a channel estimator in the feedback loop which receives information from both the channel and the LDPC decoder, and feeds back an updated channel estimate to the BCJR equaliser. The LDPC decoder in the diagram is modified to provide the higher order information that the channel estimator requires, however in theory this information could also be derived by modifying the equaliser algorithm.

Kaleh and Vallet adapted the Baum-Welch expectation-minimisation algorithm to iteratively improve the estimate of a channel [83]. The technique was later adopted by Lopes and Barry to be applied to turbo equalisation [84]. This technique examines the received symbols together with pairwise posterior probabilities of the symbols. Lopes and Barry used this technique with success, presenting results in [85].

Using the same notation as above, a string of symbols $\mathbf{s}$, with $\forall s_n \in \mathcal{A}$, where $\mathcal{A}$ is

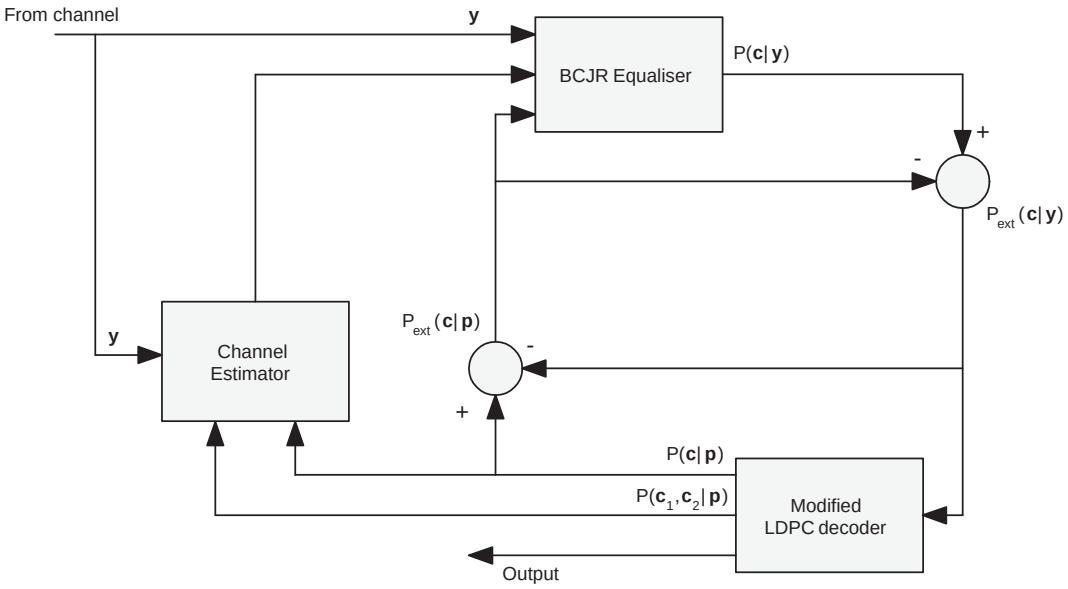


Figure 8.4: Block diagram of a blind turbo equaliser

the constellation of symbols, is sent through a channel with impulse response at time n of $\mathbf{h}_n$ and length L. The received symbol vector is denoted $\mathbf{y}$ and $y_n = \mathbf{h}_n * \mathbf{x}_n + \upsilon_n$ where ν_n is a white noise vector and $\mathbf{s}_n = [s_{n-L}, s_{n-L}+1, ..., s_n]$. A vector, $\mathbf{y}_n = [y_{n-l_1}, y_{n-l_1+1}, \ldots, y_{n+l_2}]$ is defined to denote a set of input symbols throughout which the channel response is assumed to be constant. Ideally the channel estimator would try to minimise $-\log p(\mathbf{y}_n|\mathbf{h}_n)$, which is unknown, but the likelihood function $p(\mathbf{y}_n|\mathbf{h}_n, S_n)$, where $S_n = [\mathbf{s}_{n-l_1}, \mathbf{s}_{n-l_1+1}, ..., \mathbf{s}_{n+l_2}]$ is available from the decoder as a valid alternative. The technique then begins with an initial channel estimate $\mathbf{h}_n^{[0]}$ and iteratively minimises

$$\begin{aligned}\mathbf{h}_n^{[p+1]} &= \arg\min_{\mathrm{h}} E\left(-\log p(\mathbf{y}_n|\mathbf{h}, S_n)p(S_n)\,\middle|\,\mathbf{y}_n, \mathbf{h}_n^{[p]}\right) \\ &= \arg\min_{\mathrm{h}} E\left(\sum_{i=1}^{N} |y_i - \mathbf{h}^*\mathbf{s}_i|^2\,\middle|\,\mathbf{y}_n, \mathbf{h}_n^{[p]}\right)\end{aligned} \tag{8.1}$$

which leads to the following set of normal equations to find $\mathbf{h}_n^{[p]}$ [83]

$$\mathbf{R}_n^{[p]}\mathbf{h}_n^{[p+1]} = \mathbf{r}_n^{[p]} \tag{8.2}$$

where

$$\mathbf{R}_n^{[p]} = \sum_{i=n-l_1}^{n+l_2} E\left(\mathbf{s}_i \mathbf{s}_i^* | \mathbf{y}_n, \mathbf{h}_n^{[p]}\right) \tag{8.3}$$

$$\mathbf{r}_n^{[p]} = \sum_{i=n-l_1}^{n+l_2} y_i^* E\left(\mathbf{s}_i | \mathbf{y}_n, \mathbf{h}_n^{[p]}\right). \tag{8.4}$$

Equation 8.4 requires first order (mean) statistics, which are easy to extract from the output of a conventional MAP decoder such as for LDPC, which outputs code bit probabilities $P(c_k|\mathbf{y})$, which can be converted to symbol probabilities $P(x_k|\mathbf{y})$ using a bit-to-symbol mapper. From these probabilities the expectation term used in equation 8.4 can be evaluated

$$E\left(\mathbf{s}_k | \mathbf{y}_n, \mathbf{h}_n^{[p]}\right) = \sum_{a \in \mathcal{A}} a P(s_k = a|\mathbf{y}). \tag{8.5}$$

Equation 8.3 requires second order (covariance) statistics. Evaluating these generally requires modifications to the decoder such that pairwise probabilities, $P(c_k, c_l|\mathbf{y})$, be output. After converting these from bit to symbol probabilities, $P(x_k, x_l|\mathbf{y})$ the expectation term from equation 8.3 can be evaluated as

$$E\left(s_k, s_l^* | \mathbf{y}_n, \mathbf{h}_n^{[p]}\right) = \sum_{a \in \mathcal{A}} \sum_{b \in \mathcal{A}} a b^* P\left(s_k = a, s_l = b|\mathbf{y}\right). \tag{8.6}$$

For the above equations, if the channel is assumed to be invariant throughout the entire transmission of N samples, then setting $l_1 = n - 1$ and $l_2 = N - n$ will yield a single channel estimate.

8.3.1 Generating pairwise probabilities

Work was performed on the problem of generating pairwise probabilities for equation 8.6 by Gunther et. al. who proposed modifications to the BCJR algorithm [86] or the LDPC decoder [87]. In [87] a 'generalised' LDPC decoder was derived, which adds steps to the standard LDPC BP decoder (see section 7.5.4) that evaluate and output the required probabilities.

Vertical step: in addition to the steps covered in section 7.5.4, the following is metric

is computed (keeping the same notation)

$$\delta r_{m(n_1 n_2)} = \prod_{n' \in \mathcal{N}_m \setminus \{n_1, n_2\}} \delta q_{mn'} \tag{8.7}$$

$$r^0_{m(n_1 n_2)} = \frac{1}{2}(1 + \delta r_{m(n_1 n_2)}) \tag{8.8}$$

$$r^1_{m(n_1 n_2)} = \frac{1}{2}(1 - \delta r_{m(n_1 n_2)}). \tag{8.9}$$

which computes pairwise r-messages for every combination of n_1 and n_2.

Horizontal step: in addition to the steps covered in section 7.5.4, using the r-messages computed in the above mentioned step, pairwise q-messages are generated by

$$q^{x_{n_1}, x_{n_2}}_{\mathcal{M}_{n_1} \cup \mathcal{M}_{n_2}} = \alpha P(c_{n_1} = x_{n_1} | \mathbf{y}) P(c_{n_2} = x_{n_2} | \mathbf{y}). \prod_{m \in \mathcal{M}_{n_1} \cup \mathcal{M}_{n_2}} r^{x_{n_1} + x_{n_2}}_{m(n_1 n_2)} \tag{8.10}$$

where α is a normalisation constant chosen so

$$q^{0,0}_{\mathcal{M}_{n_1} \cup \mathcal{M}_{n_2}} + q^{0,1}_{\mathcal{M}_{n_1} \cup \mathcal{M}_{n_2}} + q^{1,0}_{\mathcal{M}_{n_1} \cup \mathcal{M}_{n_2}} + q^{1,1}_{\mathcal{M}_{n_1} \cup \mathcal{M}_{n_2}} = 1. \tag{8.11}$$

These values are the pairwise pseudo-posterior probabilities required for equation 8.6, which completes the blind equaliser.

The extra steps required for the LDPC decoder increase the processing time by a considerable amount. In practice as only a limited number of pairwise probabilities, those between all indices n_1 and n_2 that fulfil the constraint $|n_1 - n_2| \leq L$ need to be decoded, which limits the increase somewhat.

8.3.2 Semi-blind turbo equalisation performance

The results presented in section 12.3 show turbo equaliser performance under AWGN and CCIR channels, using a small amount of training data.

8.3.3 Justification for inclusion in the proposed system

Turbo equalisation is a modern technique which is certainly worth consideration for use in any modern communications system. It requires a large amount of computing power to implement, however, so it may be difficult to integrate into a real-time bidirectional system. As the system being developed for this project is primarily one way, not real time and with a large amount of computing power at the receiver, it is well suited for implementing a technique such as this.

Combining the blind turbo equalisation technique above with a little training data to provide the initial channel estimate and discipline the equaliser during the course of the transmission can be termed semi-blind turbo equalisation. The main reason to use semi-blind turbo LDPC equalisation is that it theoretically reduces the amount of training data required to be sent with the transmission. The packet structures of the HF standards described in chapter 3 all require a substantial amount of training data to be sent with the data payload. Using semi-blind turbo equalisation has the potential to reduce the amount of training data down to much lower levels for similar error rates, as evidenced by the evaluation in section 12.3. It is suspected (but not tested as part of this project) that semi-blind turbo equalisation should outperform more established techniques when integrated into the same standard waveform.

Chapter 9

Antennas

The antenna in a radio communication device refers to the element used to radiate RF power from the transmitter, and to detect and collect radio transmissions from the atmosphere into the receiver electronics. This chapter describes this project's particular requirements for antennas, the design challenges and ramifications for the rest of the system. It postulates a buried antenna as a suitable model for the link budget, and provides realistic Figures for the performance of buried antennas, of which there is currently very little in the literature. Further to this the design of some buried antennas is described and details of the evaluation and characterisation procedure and their performance results are given.

9.1 Antenna requirements

There will be two antennas used in the final system, one for the base station and one for the remote unit. Each will be used to both transmit and receive signals from and to their respective devices. As explained in section 4.3 the antenna connected to the base station is assumed to be a well matched, directional antenna, such as a log-periodic type. Such antennas are already well known and documented [88] and as such merit little further attention. The antenna at the remote unit however, as mentioned in section 4.2.4, is assumed to be of poor match and gain. Further assumptions have to be made about this arbitrarily poor antenna, and it was decided that a buried antenna might be a suitable model around which to base the design of the system.

The antenna specification of the remote unit has implications for the design of the unit's electronics, especially the radio frequency (RF) front end transmission circuitry. The power amplifier output stage must be resistant to degradation of the signal and damage to the physical components caused by signals reflected back from the antenna in worst case scenarios where the match between the antenna and amplifier is very poor. The interface between the remote unit and the transmit antenna can be implemented using an antenna tuning unit (ATU), a device that automatically corrects for the mismatch between the modem transmitter output and the antenna. These devices are covered briefly in section 10.2.2 but it was decided that designing one was outside the scope of this project.

9.1.1 Matching and efficiency

As stated above the object of a transmit antenna is to radiate as much power as possible in the direction of the receiver. A receive antenna attempts to collect as much energy as possible and transfer it to the receive amplifier input. An antenna, operating at a given frequency has a characteristic impedance, much the same as that seen at the terminals of an resistor-inductor-capacitor (RLC) filter [88]. They have resonant frequencies around which the characteristic impedance of the antenna drops to a lowest value. It is desirable to transmit RF signals close to this resonant frequency, where the impedance is effectively resistive, to minimise reflective power losses in the system and maximise the radiated power. It is a well known phenomenon in RF and transmission line electronics that to ensure maximum power transferal, from the output of one network to the input of another, the characteristic impedance at the former's output must be equal to the characteristic impedance of the latter's input [89]. This phenomenon forms the basis of 'impedance matching' techniques, where electronics networks are designed to give a standardised impedance at their terminals; this impedance is typically 50Ω for RF circuits. If impedances between two networks are not matched, power transferal between the two is sub-optimal and power will be reflected back from the input terminal of one network into the output terminal of the other. This can be an especially serious problem when using powerful transmit amplifiers with poorly matched antennas, where power reflected back into the output of the amplifier can cause problems and potentially cause permanent damage to the amplifier electronics.

Impedance matching networks can be constructed with different output and input

impedances. When such networks are connected to the output terminal of another network which matches the input impedance of the impedance matching network, the output impedance of the ensemble is equal to the output impedance of the impedance matching network. These circuits, in practice, are designed such that the input (or output) impedance of the network match the output (or input) impedance of an existing circuit, the output impedance of the circuit is thus *transformed* to the designed output impedance of the matching network. Such networks are of great use when it is required that power transfer is maximised between a circuit designed for a given characteristic impedance (for example 50Ω) and a device with an arbitrary characteristic impedance which can not be easily changed.

To integrate an arbitrary antenna into a system, the resonant point of the antenna must first be determined and the impedance at this point measured. This will probably not match the characteristic impedance of the circuit to which the antenna will be connected, and therefore an impedance matching network is designed to match the two different impedances together. This will maximise power transfer between the antenna and the radio electronics, and reduce reflected power to a minimum. There will be a power loss associated with the matching network, but this will be small compared to the losses seen if two unmatched circuits are simply connected together. This process is normally accomplished in practice using an off-the-shelf ATU [90].

9.1.2 Voltage Standing Wave Ratio

In any transmission line or antenna, when its input is excited by a source, there is the possibility of power being reflected back as a result of discontinuities in the line. In terms of the efficiency of an ideal antenna, any input power that is not radiated from it will be reflected back to the power source used to drive it. The ratio of voltage resulting from power reflected back from it to that which is supplied to it is known as the voltage ratio

$$\Gamma = \frac{V_r}{V_s} \tag{9.1}$$

where v_r and v_s are the voltages reflected from and supplied to the transmission line or antenna. The value of Γ is a complex number arising from the complex vectors used to represent the magnitude and phase of the voltages. If $\Gamma = 0$ the impedance of the input of the transmission line is the same as the output of the voltage source and there is no reflected voltage. This is known as an *impedance matched* system

and is the aim for most radio communication systems as it ensures maximum power transfer from the source to the destination.

The voltage standing wave ratio (VSWR) is a real value used to simplify the application of Γ. It is defined as

$$\text{VSWR} = \frac{1-\rho}{1+\rho} \tag{9.2}$$

where $\rho = |\Gamma|$ is the modulus of the voltage ratio. This value is constrained by VSWR≥ 1, where VSWR=1 indicates a perfect match between the input of a transmission line and the output of the power source, this is the ideal case.

The VSWR is an important metric as it is a simple way to determine the efficiency of an antenna or transmission line. As mentioned before the VSWR is ideally as close to 1 as possible, which is the point at which $|\Gamma|$ becomes 0 (i.e. there are no reflections, and the match is perfect). Deviations from this ideal characteristic will lead to lower power transfer from source to load and inefficiency. Also, higher values of VSWR can cause problems with many power amplifiers as reflected power can cause damage the output stages.

9.1.3 Mismatch loss

Mismatch loss is the loss of output power compared to input power from imperfect matching of an electrical connection and reflections. It is related to VSWR using the following relationship [88]:

$$L_{m,dB} = 10.\log_{10}\left(1 - \left(\frac{\text{VSWR}-1}{\text{VSWR}+1}\right)^2\right) \tag{9.3}$$

Figure 9.1 shows a graph of matching loss plotted against a range of values of VSWR.

These results provide a metric relating the VSWR Figure of a poor antenna at a given frequency to the loss of power at the output. They do not take into consideration signal absorption effects for antennas buried beneath the ground surface, which will be discussed in the following section.

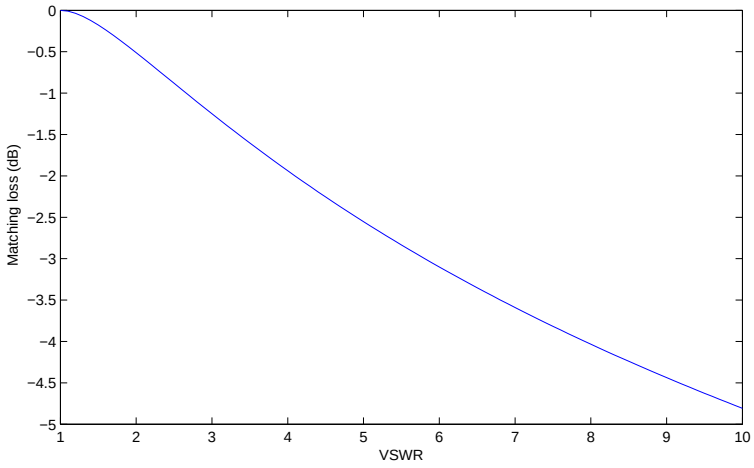


Figure 9.1: Matching loss plotted against VSWR

9.2 Buried Antennas

The nature of the proposed system may necessitate the use of a low visual impact or buried antenna at the remote unit for transmission of data and reception of commands and repeat requests from the base unit. Even if this turns out not to be a stringent requirement, a buried antenna might provide a useful model of a poor antenna from which to draw rough estimates of the expected performance characteristics of the type of antenna which will be used in this type of project. Notwithstanding any defence application, buried antennas are also a worthy subject for investigation due to the general uses of such devices in other endeavours. For communication with remote units in any geographically difficult to access locations or environmentally difficult or hazardous environments such as near volcanos, areas of high seismic activity, high winds or polar regions, it might be undesirable to attempt to erect an antenna mast for a number of reasons. Furthermore, in national parks, protected areas or areas of great natural beauty it may be undesirable or illegal to erect antenna masts. In such cases burying the antenna may be the only feasible alternative.

There is little publicly accessible research available that has been performed in the area of buried or submerged antennas, due to the inherent sensitivity of many of their applications. There is, however, some buried antenna research conducted during the Cold War which has been published [91][92][93][94]. Since that time most research in this area has been performed mostly by the amateur radio community. Buried antennas have been used by radio amateurs for a wide range of reasons including as dummy loads [95] and practical systems where it is impossible to erect

an antenna in a back yard [96].

When an antenna is properly conFigured and suspended in air, the wavelength of a signal moving through it is given by $\lambda \approx \lambda_0 = c/f$, where λ is the wavelength of the signal, λ_0 is the wavelength in free air, c is the speed of light and f is the frequency in Hz. Where there is an appreciable difference in permittivity and conductance of the surrounding medium to that of free space, the wavelength of the signal is given by the following equation [97]:

$$\lambda = \frac{\lambda_0}{\left[\epsilon_r^2 + \left(\frac{\sigma}{2\pi f \epsilon_0}\right)^2\right]^{1/4}} \tag{9.4}$$

where ϵ_r is the relative permittivity of the surrounding medium (see table 9.1), $\epsilon_0 = 8.854 \times 10$ is the permittivity of free space and σ is the conductivity of the soil in S/m. The relative permittivities of common soil types are given below in a table reproduced from [5].

Material	**Dielectric Constant**
Sand (dry)	3-6
Sand (saturated)	20-30
Silts	5-30
Shales	5-15
Clays	5-40
Humid soil	30
Cultivated soil	15
Rocky soil	7
Sandy soil (dry)	3
Sandy soil (saturated)	19
Clayey soil (dry)	2
Clayey soil (saturated)	15
Sandstone (saturated)	6
Limestone (dry)	7
Limestone(saturated)	4-8
Basalt (saturated)	8
Granite (dry)	5
Granite (saturated)	7

Table 9.1: Permittivities of different soil types [5]

Equation 9.4 shows that surrounding an antenna within a dielectric will have the effect of reducing the wavelength of the signal for a given frequency, effectively reducing the speed of light (more precisely the electromagnetic signal) through it.

It was decided that it would be useful to evaluate the performance of some buried antennas for the purposes of this project, to validate and build upon the limited data currently available in the literature.

9.2.1 Buried antenna construction

Three antennas were constructed, two being single ended types of 5 m and 10 m lengths, and one being a 10 m centre-fed dipole which was constructed by laying two 5 m lengths end to end. The antennas were constructed from 1.5 mm multi-core wire with a plastic sheath. 10 m was decided as the maximum length for the antenna tests as it quickly becomes impractical and time consuming to bury lengths longer than this.

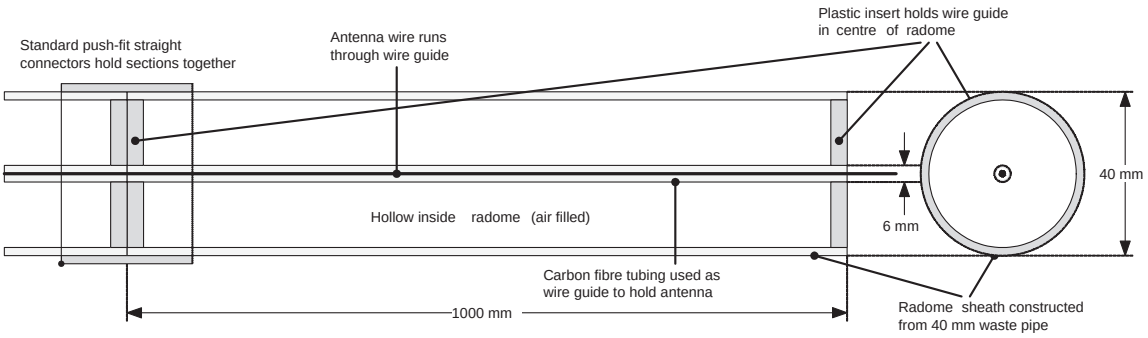


Figure 9.2: The construction of a segment of radome

Given recommendations in the literature [96], it was decided that a radome should be constructed to house the buried antennas. Ten 1m sections of radome were constructed that could be attached together to form a radome of various lengths. Each section was constructed with an outer sheath which defined the diameter of the radome and a thinner inner tube which was used to hold the antenna wire in the middle of the radome. The outer sheath was constructed from lengths of 40mm diameter polyurethane waste pipe and the inner tube was constructed from 5mm diameter (4mm inner diameter and 6mm outer diameter) carbon fibre tubing. The inner tube was held in the centre of the outer sheath using two plastic inserts, glued in position at either end of the tubes. The tubes were held together with standard straight 40mm push-fit plastic connectors, which allowed easy reconfiguration of the radomes in the field. Figure 9.2 shows a diagram of the radome tubes.

9.3 Antenna tests

The three antenna configurations were tested at a beach in Hoylake on the Wirral peninsula in the north west of the UK. The object of each test was to obtain the VSWR and characteristic impedance for each configuration over a range of frequencies representative of the HF spectrum. Of these, the lower frequencies are more likely to be available for short range, NVIS links, which the proposed system will likely implement.

The beach itself was sand, which was moist and became rather oily approximately 20 cm down from the surface. The antenna radome was buried in a trench approximately 30 cm from the surface. Readings of the ground conductivity were taken, at the surface of the undisturbed sand before digging the trench. These values ranged from 5.89 to 9.39 mS/m. At the bottom of the trench, where the sand was more compact and had a greater oil content, these results were slightly higher, ranging from 6.87 to 10.69 mS/m. Additionally, the ground conductivity was measured after burying the radome and recovering. Their values were found to be slightly lower, presumably because the sand was less compact than before, and ranged from 3.6 to 9.7 mS/m.

The locations were chosen to be representative of how such an antenna might be deployed in the field, under possible tight constraints on the time available to deploy an antenna and include configurations with and without the radome, on the surface and underground. It was not possible to set up control tests with properly conFigured antennas held a suitable distance above the ground due to the difficulties associated with setting up masts in the locations chosen for testing. But as the performance of such antennas is very well known and documented [88] this was not seen to be a major problem. Each of the three antennas were buried within the radome, additionally the 10 m dipole antenna was tested with the radome resting on the surface of the sand, with the bare wire resting on the surface of the sand, and with the bare wire lightly covered in approximately 1 cm of sand. The antenna terminals were brought up from under the ground to the surface to provide test points for measurement and the single ended antennas were deployed with a ground spike consisting of approximately 30 cm of copper tubing for the tests conducted on the single ended antennas, attached to a wire of similar diameter as the earth wire using a steel nut and bolt.

Ideal single ended antennas of 5 and 10 metres in length have full wave resonant

frequencies of approximately 30 MHz and 15 MHz respectively, with half-wave ($\lambda/2$) resonant points at approximately 15 MHz and 7.5 MHz respectively and quarter-wave ($\lambda/4$) resonance points at approximately 7.5 MHz and 3.75 MHz respectively. While some of these resonant points are high compared to the likely frequency ranges used, antenna theory predicts that these resonant points occur at lower frequencies when the antenna is buried, due to the increase in dielectric permittivity and conduction of the surrounding materials [97][96], and this has been confirmed experimentally [96][93][92]. Thus the $\lambda/2$ and $\lambda/4$ resonant points are of most interest. An ideal single ended antenna approximates an ideal dipole of twice the length, as the ground plane has the effect of simulating the other half of the dipole.

A handheld SWR meter was used, which gives metrics for VSWR and characteristic impedance for frequencies set within the meter. This meter is an inexpensive device marketed at hobbyists, but was deemed sufficiently accurate to perform measurements.

9.3.1 Test methodology

For each test, the antenna to be tested was connected to the antenna terminal and the earth rod was connected to the antenna connector's earth sheath on the meter. Points of interest were identified by scanning through the range of frequencies. These points of interest were resonant points where the VSWR of the antenna fell to a 'useable' level of less than approximately 2.5. Around these points frequencies were tested and VSWR, resistance and reactance were recorded in 0.5 MHz intervals. Where the antenna's VSWR exceeded a value of 4 readings were not taken, as at these high levels the antenna is of limited use and the accuracy of the readings from the VSWR meter may be called into question (the analogue and digital readouts of the meter gave significantly different results). The meter was not capable of determining whether reactance was capacitive or inductive. The test results are available in appendix H but their VSWR graphs are shown and discussed in the following section.

9.4 Buried antenna test results

Figure 9.3 shows the VSWR characteristics of the three antennas (10 m single ended, 5 m single ended and 10 m dipole) buried within the radome at a depth of approximately 30 cm in sand with a conductivity of 5-10 mS. For the single ended antennas, these results exhibit the $\lambda/2$ and $\lambda/4$ resonant points of the antennas and for the dipole this shows the $\lambda/2$ and full-wave resonant points. A discussion of these results will be presented in the next section.

Figure 9.4 shows the VSWR characteristics of the 10m dipole in the four previously mentioned configurations (buried in the radome, laid on the surface in the radome, buried 1cm down without the radome and laid on the ground without the radome). These results show the half-wave resonant points for the all the dipole configurations and the full-wave resonant point of the dipole buried within the radome. These results are discussed in the following section.

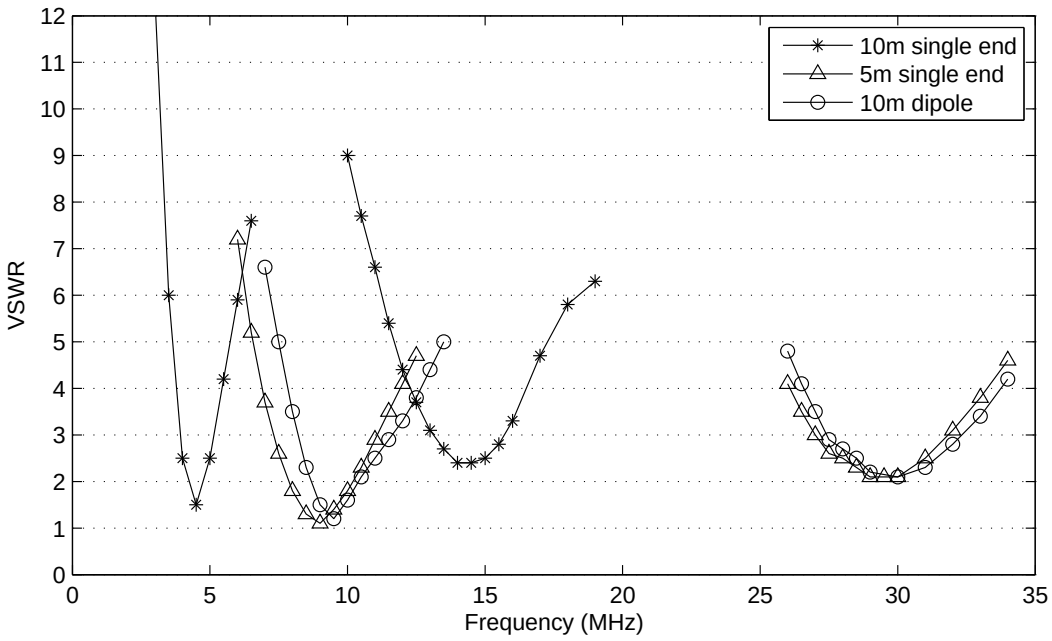


Figure 9.3: VSWR of three buried antennas

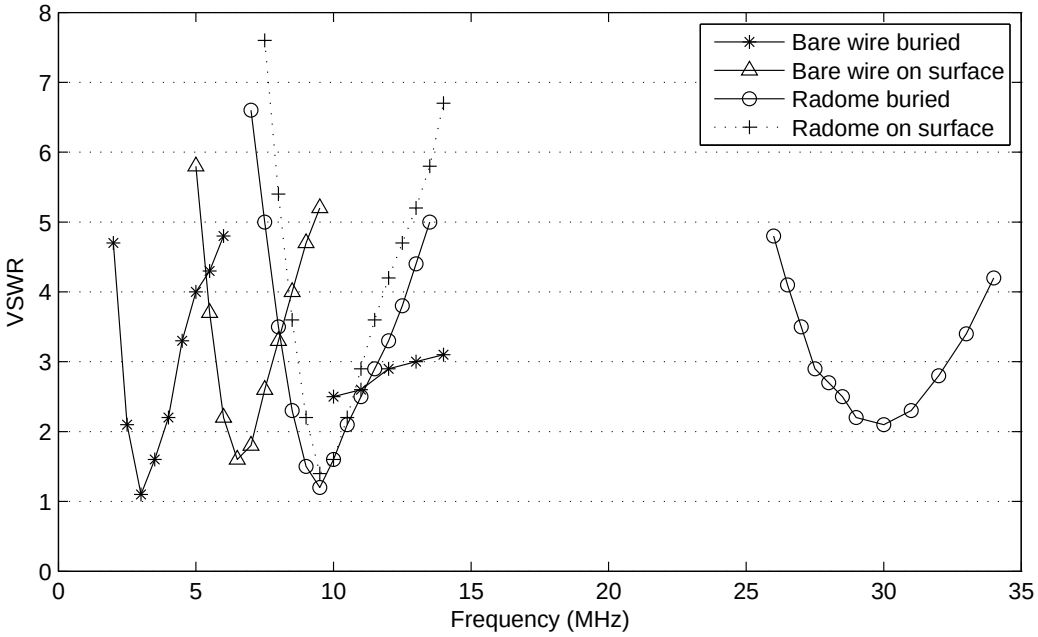


Figure 9.4: VSWR of four configurations of a 10m dipole

9.4.1 Analysis of results

Figure 9.5 is a graph generated from equation 9.4 that shows the theoretical wavelengths of signals at various frequencies when surrounded by a dielectric of relative permittivities (ϵ_r) and conductances (σ) given by table 9.1 and measurements from the soil conductivity meter respectively. Three curves are shown with values close to those measured at the scene and inferred from the table (the sand was seen to be wet but not saturated).

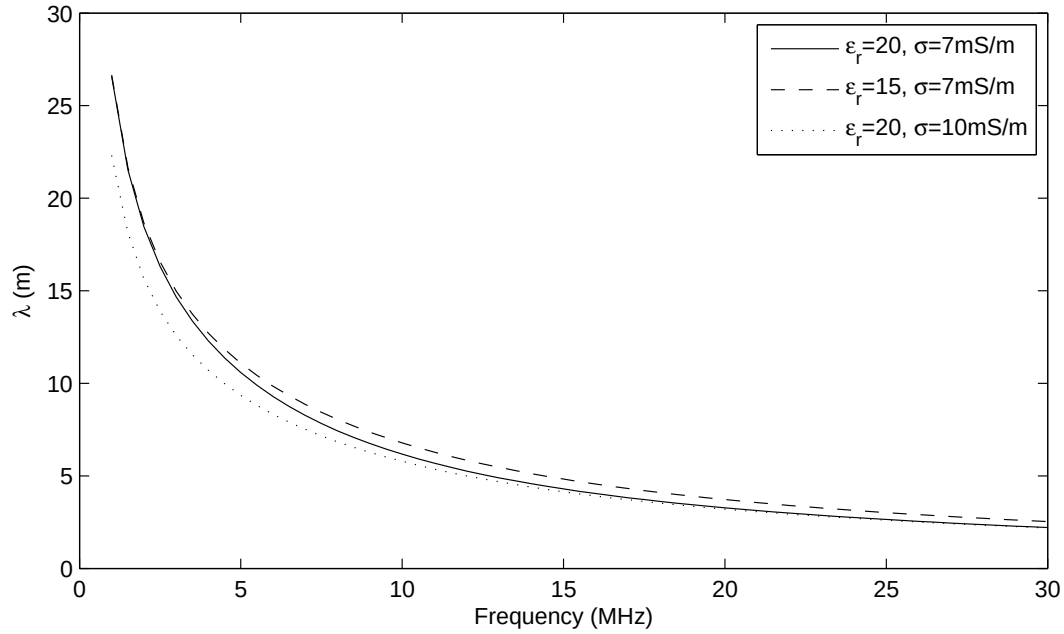


Figure 9.5: Theoretical wavelength for antennas buried in sand

From Figure 9.5, it can be seen that the full-wave resonant frequencies of a 10 m and a 5 m antenna are approximately 6MHz and 14.5MHz respectively. The reciprocal relationship between the wavelength and frequency of a signal is also broken, as the frequency decreases the wavelength increases at a slower rate than in the free space case. This is an interesting result and will be discussed with reference to the buried antenna results below.

Examining the graph in Figure 9.3 the buried antennas all show resonant points at a lower frequency than would be expected for a line in free space, akin to a properly suspended antenna. This is consistent with the theoretical results from equation 9.4 and the graph in Figure 9.5. The $\lambda/2$ resonant points for the 10 m and 5 m single ended antennas are approximately 4.5 MHz and 9 MHz respectively,

which are consistent with each other in terms of the fact that the 10 m antenna should be expected to have resonant frequencies of half of those of the 5 m antenna. Likewise the $\lambda/4$ resonant frequencies for the 10 m and 5 m single ended antennas are approximately 14.5 MHz and 29.5 MHz respectively, which are consistent with each other. The $\lambda/4$ frequencies, compared to the $\lambda/2$ frequencies, are higher than one would expect for an antenna in free space, as they are more than twice the $\lambda/2$ frequencies in each case. This however can be explained by the theoretical results for antennas radiating within a dielectric of significantly different permittivity and conductivity to free space, as described by equation 9.4.

From the graph in Figure 9.5, the theoretical and $\lambda/4$ frequencies for 10 m and a 5 m antennas radiating within a dielectric material with similar properties to the sand in which the buried antennas were tested are expected to be approximately 6/4=0.67 MHz and 14.5/4=3.625 MHz respectively. The graph in Figure 9.3 shows $\lambda/4$ resonant frequencies of significantly higher than these, which are to be expected as the antennas are radiating within a radome and reasonably close to the surface. The 10 m dipole has results very close to the 5 m single ended antenna as expected, giving similar resonant frequencies and VSWR values.

In all the test cases, the $\lambda/4$ resonant frequencies give better VSWR values than the $\lambda/2$ ones. This shows a propensity to better performance for buried antennas at the lower end of the frequency range. This may be due to the fact that higher frequencies are more effectively absorbed by a conductive medium than lower frequencies. This higher level of absorption serves to reduce the near field radiation efficiency of the antenna and subsequently reduce the quality of the match.

A number of interesting conclusions can be drawn from the graph shown in Figure 9.4. The leftmost curves are assumed to be the $\lambda/2$ resonant frequencies for the 10m dipole antenna, and can be seen to become lower as the permittivity and conductivity of the surrounding medium increases. The antennas in the radomes have the two highest resonant frequencies, at approximately 9 MHz for both the buried and surface types. The bare wire showed markedly different resonant frequencies between when laid on the ground and ‘buried’ under a thin (roughly 1 cm) layer of sand. The resonant frequencies for these cases are approximately 6.5 MHz and 3 MHz for the antenna laid on the ground and the antenna buried under the surface respectively.

9.4.2 Takeoff angle

The takeoff angle of the lobes of the transmit antenna is of importance in a practical system as it is desirable that the signal is radiated with maximum gain in the elevation and azimuth required to intercept the receiver via the reflection point(s) in the ionosphere.

The angle of attack of the main radiating lobes of the antenna were not measured, but inferences can be drawn on the likely takeoff angle of the main lobes from two properties of the antenna. The fact that the antenna is buried in a dielectric may have the effect of lowering the takeoff angle of the lobes. This is due to refraction of the electromagnetic signals as they traverse the ground-air interface. The incident angle to the interface in a medium of high permittivity is closer to the normal than the incident angle on the other side of the interface, in the medium of low permittivity. This behaviour is described by Snell's law which is given as [98]

$$v_g \sin\Theta_g = v_a \sin\Theta_a \tag{9.5}$$

where v_g and $v_a = c$ are the velocities of an electromagnetic wave in the ground and the air respectively, Θ_g and Θ_a are the incident angles in the ground and the air respectively.

If the ground is lossy, then it is obvious that longer paths through the ground will see greater attenuation. This means that signals radiated from the antenna will be more attenuated the further they deviate from the normal to the ground's surface. This will have the effect of raising the takeoff angle of the main lobes of the transmit antenna.

The other property of the antenna that affects the takeoff angle is the antenna height. As a dipole is lowered towards the ground the takeoff angle of the main lobe rises. A simulation was conducted using the EZNEC software [99] for a 7.5 m dipole radiating a signal of 10 MHz ($\lambda/4$). Figure 9.6 shows elevation plots of the radiated far-field strength of this antenna suspended above the ground by a full wavelength (30 m), a half wavelength (15 m) and a quarter wavelength (7.5 m) in red, blue and green respectively. The ground's relative permittivity was set as $\epsilon_r = 16$ with a conductivity of 1mS/m, which is representative of a 'sandy loam' soil type. It is apparent from these plots that the take off angle of the main lobe rises the closer the dipole is brought to the earth. Unfortunately the software is not capable of modelling a buried antenna and the plots become inaccurate at positions very close to the ground so it

was not possible to perform these simulations. However it can be inferred from the results in Figure 9.6 that the takeoff angle of the main lobe of a buried antenna will be close to the zenith.

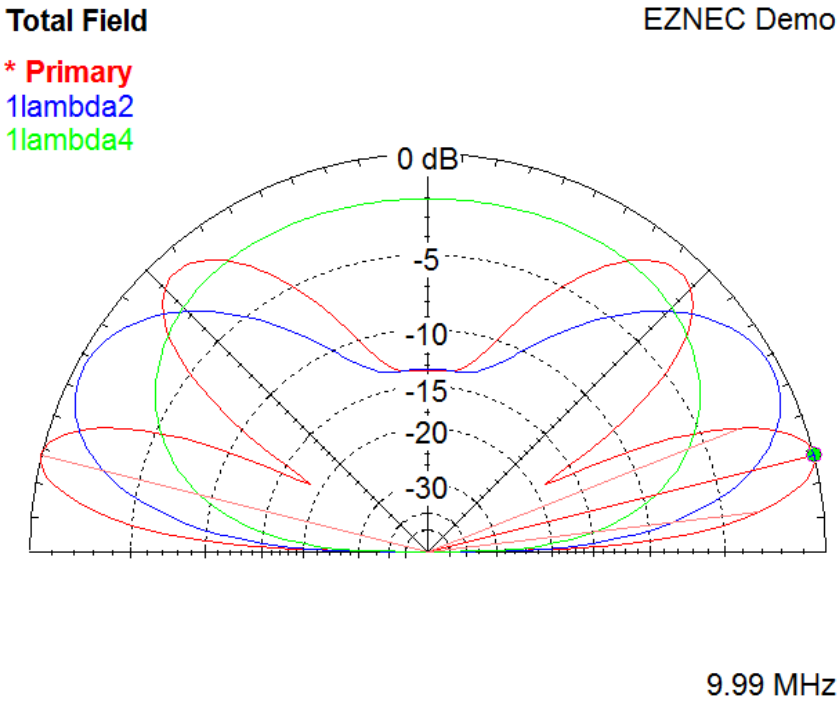


Figure 9.6: Elevation plot of dipole at various heights (see text for plot descriptions)

Unfortunately these results are inconclusive as to the actual radiated field strength of a buried dipole, and the conclusions are rather speculative. It would certainly be worth conducting a practical measurement of some such antennas as part of any further work.

9.5 Antenna Recommendations

The buried antennas discussed in this chapter have been shown to have good performance at the lower end of the frequency range, where near vertical incidence skywave (NVIS) communications are most likely to take place. The fact that a thin layer of dirt covering a bare wire on the ground has such a profound effect on the resonant frequency of the antenna makes it impractical for use within a HF system. This type of antenna will be very susceptible to wind, which may blow dirt onto it, and rain, which may substantially affect the dielectric properties of the surrounding medium. If the VSWR is liable to change as much as the results suggest, the resultant poor match could impede transmission of the signal and damage to the power

amplifier from reflected signals from the antenna could also result. For this reason it is recommended that a radome be constructed in a similar manner to the construction used for this project, the details of which are given above in section 9.2.1.

Further work is required to determine the suitability of a buried antenna for a short range, NVIS system, but some of the analysis in the previous suggestion suggests that such an antenna may be well suited to such an application.

Chapter 10

Implementation of the remote unit

This chapter will describe the design and implementation of the remote unit, which forms one half of the communications link that this project is tasked to design. The *remote unit* is the device to be deployed remotely, which will transmit information to the base station via the HF channel. It is to be placed at a site of interest, geographically distant from the receiver and possibly in a remote or difficult to access place. Data transfer will primarily take place from the remote unit to the base station.

Section 4.2 gives the requirements of the remote system which are summarised here:

- **Size:** Physically Small.
- **Power Source:** Low power operation from 4 alkaline D cells.
- **Operating schedule:** Opportunistic, with a minimum operating period of 1 week and 1 month being highly desirable.
- **Antenna:** Operating with inefficient antenna (short or even buried wire).
- **HF waveform:** Designed for low probability of detection.
- **Data:** Five A4 pages of (pre-compressed) text per day.

This device was designed and implemented as a proof-of-concept for the transmitter unit. As described in the requirements summarised above, the most important property of the remote unit hardware is that it consumes very little power. The power

requirements are split into that which will be used to amplify the signal prior to transmission, and the power required to run the other subsystems such as signal processing. The very low power requirements of the board dictated every aspect of its design, with low power components used throughout. It was desired to evaluate the possibility that such a device could perform the complicated encoding procedures required of the LDPC-based techniques described in chapter 8 as well as performing the operations required of a digital modem.

10.1 Remote unit hardware

The transmitter hardware is required to implement a HF local oscillator, a transmit chain for transfer of data to the base station, a receive chain for receiving instructions and repeat requests from the base station and a processing element for performing digital signal processing (DSP) and scheduling the functions of the board. A printed circuit board (PCB) was designed and manufactured to provide these functions, which is described below. The PCB was designed in Altium Designer and the files are given in Appendix E. The schematics are also reproduced later in this chapter in section 10.3. A block diagram giving an overview of the functionality of the PCB is given in Figure 10.1, and the main functions are discussed in more detail later in this chapter.

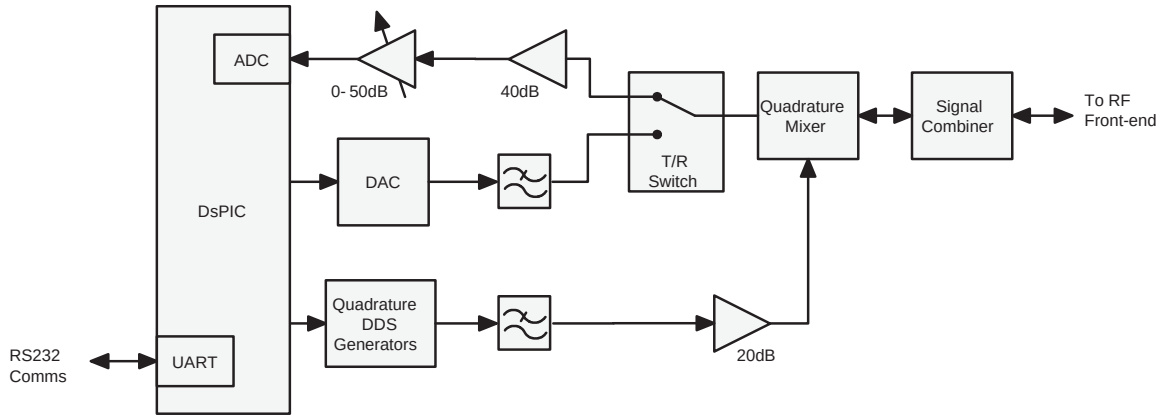


Figure 10.1: Overview of the functionality of the remote unit board.

Figure 10.2 shows a picture of the PCB designed and manufactured during the course of this project. At the top left of the board the voltage regulators for the eight rails can be seen near the DC input port. The top right of the board is the RF front end, with the metal-cased mixers and power splitter/combiner. The top centre

of the board is the receive chain and below this the transmit chain is located. The DSP microcontroller is located at the bottom left of the board and to the right of this the DDS local oscillator and anti aliasing filter can be seen.

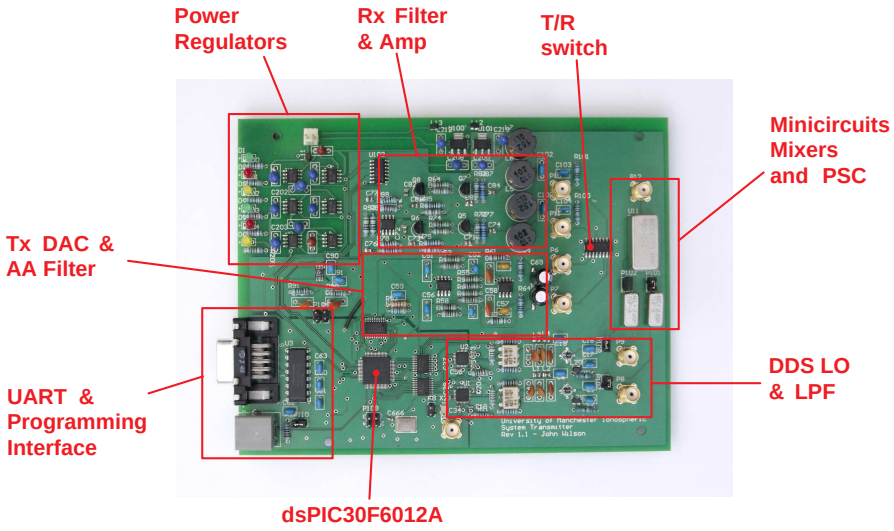


Figure 10.2: The remote unit PCB

10.1.1 Processor

The processor is responsible for managing PCB resources, passing data between devices, encoding the data with using a quasi-cyclic low density parity check (QC-LDPC) forward error correction (FEC) code (see sections 7.5.2) and performing DSP operations on the input and output signals. The required DSP operations are listed below:

- Phase shift keying (PSK) modulation is performed on an input stream of data to be transmitted. For justification of the choice of this technique see section 6.4.
- A root-raised cosine (RRC) interpolating filter is required to filter the output signal, this up-samples the modulated transmit signal to the digital to analogue convertor (DAC) sampling frequency and filters it. RRC filters aid clock recovery at the receiver [3].
- A Costas loop [100] is required at the output of the receive chain analogue to digital convertors (ADCs) to lock to the received signal's carrier, and compensate for Doppler effects.

It was decided that a Microchip dsPIC device be used for the main processor element due to familiarity with the platform and the accessibility of debugging hardware within the University. These devices are based on the 16-bit PIC architecture, with additional instructions useful for accelerating the execution of filters and other common DSP tasks. The dsPIC30f6012A device was selected [101], which is a 64-pin thin quad-flat pack (TQFP) device with a large RAM, suitable for storing multiple data buffers in the system, and sufficient I/O pins to control the various functions on the PCB [102]. This device operates at low power, consuming approximately 200 mA with a 5 V power supply and a clock frequency of 32 MHz. An 8 MHz crystal is connected which can be multiplied with an internal phase locked loop to provide a 32, 64 or 128 MHz clock source. This allows flexibility in design but entails a trade off of power consumption with performance.

10.1.2 Local oscillator

The local oscillator in the system is required to provide two sinusoidal signals of the same frequency and 90^o phase difference, such signals are termed in *quadrature phase*. This enables so-called *quadrature/in-phase* (Q/I) signalling, which allows complex symbols to be transmitted over a real channel, enabling easy implementation of techniques such as quadrature phase shift keying (QPSK). The oscillator is further required to be programmable to output a wide range of frequencies, to support communications across the range of the HF spectrum (for discussion refer to chapter 2).

For these reasons the decision was made to use direct digital synthesis (DDS) to generate a local oscillator signal. Direct digital synthesis is a modern technique which uses a digital sine wave synthesiser in combination with a high speed DAC to generate an analogue sinusoid. The frequency and phase of the output waveform of a DDS are controlled by writing values to registers within the IC by means of a communications bus from the processing device. DDS oscillators are advantageous in that they are highly configurable and can change the phase and frequency of a signal with low latency; the latency of the communications bus is the only limiting factor. For a detailed discussion of DDS techniques refer to [103].

A block diagram of the local oscillator is presented in Figure 10.3. Two Analog Devices AD9913 DDS ICs were used to generate the oscillator signals. These are low power devices, consuming a maximum of 45 mA with a 1.8 V supply, that run at

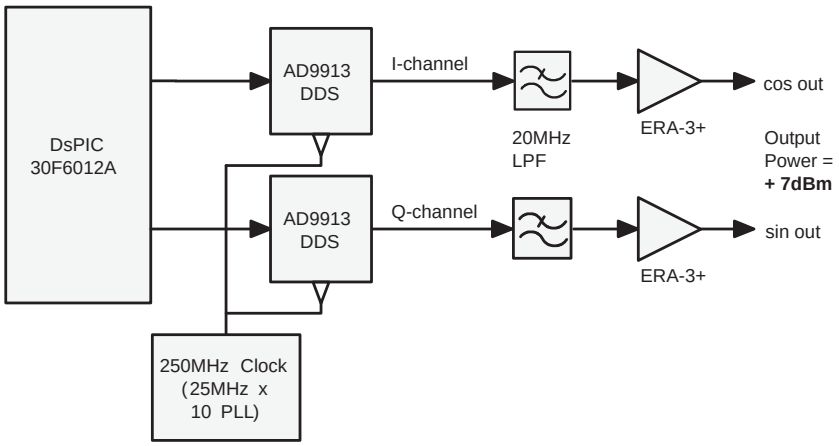
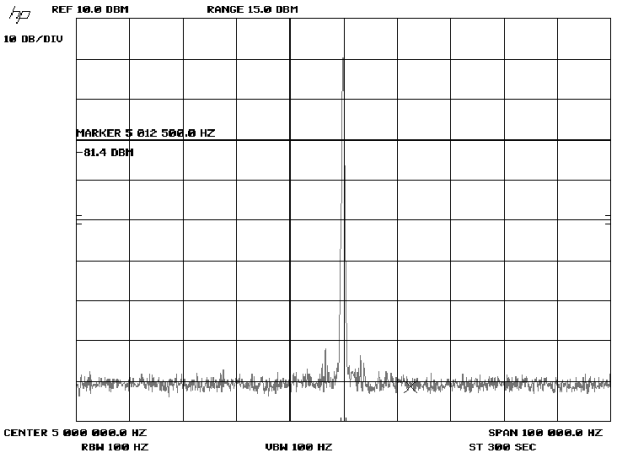
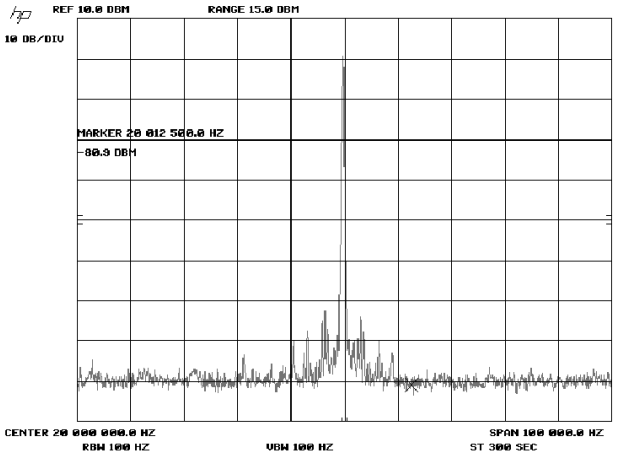


Figure 10.3: Block diagram of the local oscillator

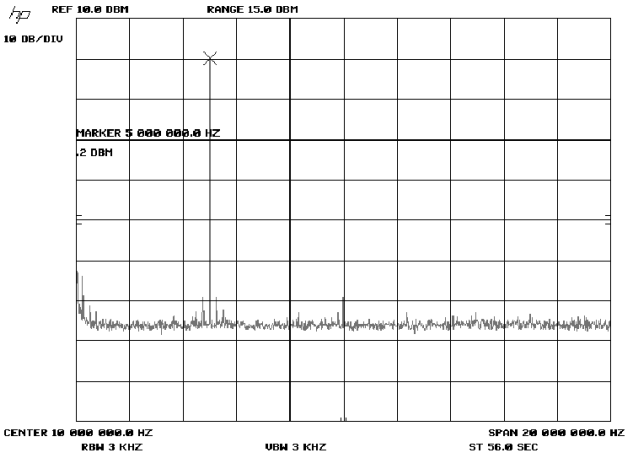
a sampling rate of up to 250 MHz [104], sufficiently high that harmonics and images over the Nyquist frequency are easily filtered out at frequencies across the entire HF spectrum. The output of each DDS generator is coupled via a transformer to a low pass, three pole LC anti-aliasing filter. The filter was designed to cut-off at 20 MHz, which limits the maximum frequency available from the LO. This decision was taken due to the fact that the system will most probably be deployed as part of a *near vertical incidence skywave* (NVIS) system which are known to use lower frequencies (< 10 MHz) [12]. The filtered signal is then fed into a Minicircuits ERA-3+ monolithic microwave integrated circuit (MMIC) signal amplifier, this is a 21 dB, 50 Ω matched general purpose amplifier block [105]. During testing it was found that the output of the amplifiers were over-driving the mixers. This was fixed by programming the DDS to a lower current (approximately 1mA p-p) by replacing the recommended 4.6 kΩ programming resistors with 15 kΩ resistors. The spectrum of one channel of the local oscillator (LO) output was observed using a Hewlett-Packard 3585B spectrum analyser. The mixer was disconnected and the output of the local oscillator amplifier was probed, thus taking into account the effects of the filter and amplifier on the output spectrum. Two frequencies were tested; 5 MHz represents the lower end of the HF band, and 20 MHz is the maximum frequency that the LO can output (as determined by the 20 MHz cut-off point of the anti-aliasing filter). For both of these frequencies, a narrow-band scan of 100 kHz and a wide-band scan of 20 MHz were conducted. The observed waveforms are reproduced in Figure 10.4. The LO output appears clean, with no spurious components of an amplitude higher than -60 dB of the fundamental frequency.



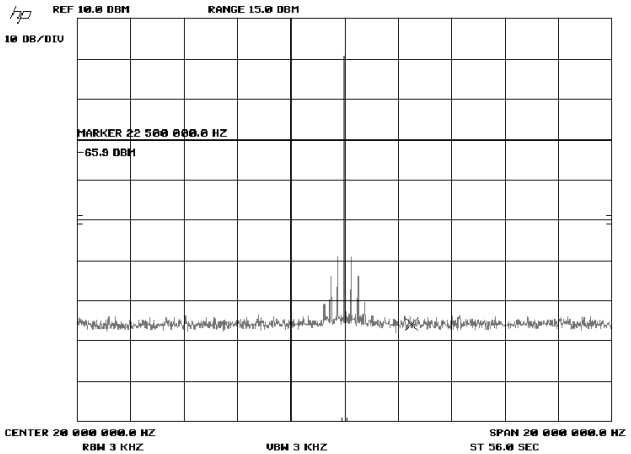
(a) LO at 5MHz, 100kHz scan



(b) LO at 20MHz, 100kHz scan



(c) LO at 5MHz, 20MHz scan



(d) LO at 20MHz, 20MHz scan

Figure 10.4: Spectrum analysis of LO output

10.1.3 Transmit chain

The transmit chain is responsible for generating the baseband transmit signal and up-mixing it to radio frequency. The planned bit rate through the transmit chain is expected to be around 1 kbps, and it is unlikely to exceed 2 kbps (see the link budget analysis in chapter 5 for further discussion). Data modulation is required to be simple, for power efficiency (see [48] and section 6.4 for discussions). The system will therefore be conFigured to employ binary- and quadrature- phase shift keying (BPSK/QPSK) to modulate the digital data onto the RF carrier. To use complex-valued symbol modulation schemes like QPSK, it is required that two channels be used; one, the *in-phase* component, holding the real part of the signal and the other, the *quadrature* component, holding the imaginary part. These are mixed with the cosine and sine carriers from the local oscillator and combined before being fed to the transmit antenna. The transmit chain on the PCB consists of a dual channel

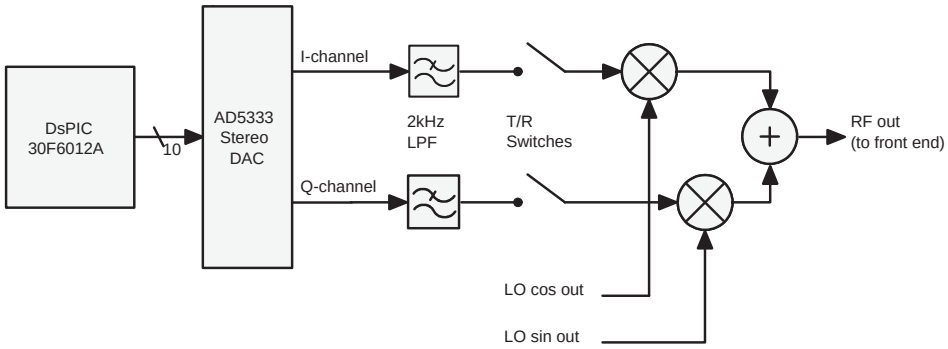


Figure 10.5: Block diagram of the remote unit transmit chain

digital to analogue convertor (DAC), a low pass anti-aliasing filter for the each output of the DAC, two mixers to multiply the baseband signals with the carriers from the LO, and a power combiner to add the signals prior to being fed to the RF front end. Figure 10.5 gives a block diagram of the transmitter unit. In addition to that mentioned above, switches are also present on each channel before the mixers. These switches reconFigure the system for transmit or receive by connecting the transmit or receive chain electronics to the mixers. The mixers and power combiner are bidirectional (with the power combiner effectively splitting the power of an input signal when operated in reverse), which allows them to be shared, and importantly simplifies the design of circuit which uses a single antenna to both transmit and receive.

The DAC chosen was an Analog devices AD5333, which is a two channel, 10-bit general purpose device [106]. The AD5333 is connected to a general purpose I/O

(GPIO) bank on the DSP via an 10-bit parallel link which allows fast transfer of output symbol data, the output value registers can be conFigured to be updated simultaneously, from values from prior transfers stored in buffers. Each DAC output is followed by a 2-pole Sallen-Key low pass filter with unitary gain and 2 kHz cut-off frequency. Both filters are implemented on a two dual channel op-amps, the input stages of both channels of the filter are based on an Analog Devices AD8667 low-noise, general purpose op-amp and the output stages of both channels are driven by a National Semiconductor LM7332 op-amp which provides a high output current (70 mA), suitable for driving the 50 Ω mixer load.

10.1.4 Receive chain

The remote unit requires some means of receiving commands and repeat requests from the base station. For this reason a receive chain has been added to the remote unit PCB. The number of types of required commands to be sent from the base station to the receiver is likely to be low and representable by a small number of basic symbols. Chirps or modulated pseudo-random noise sequences are suitable. The best way to detect symbols such as these would be to implement a simple bank of matched filters on the DSP. Additionally, little complexity is required to implement such a receiver, and this is reflected in the design of the system, which uses the ADC on the DSP controller to detect signals, instead of a dedicated, higher performance ADC. The receive chain has been implemented to use complex baseband (QI) signalling, to take advantage of the quadrature local oscillator on board the PCB.

The receive chain consists of two amplifiers attached in series for each channel. The amplifiers are a dual transistor common base amplifier which provides approximately 40 dB of gain, and a variable amplifier which gives 0-50 dB of gain. The variable gain amplifier is based around the serial combination of a $10\times$ inverting operational amplifier gain block which uses an analogue devices AD8667 dual op-amp device (one op-amp for each channel) [107] and a Linear Technologies LTC6911 dual channel variable amplifier [108], which gives $0 - 100\times$ linear gain. The common base transistor amplifier is adapted from a design found in [109], but converted to use a 5 V DC supply.

The input of the amplifiers are connected via the transmit/receive CMOS switch to the mixers. Filtering and matching to the 50 Ω load required by the mixers is

achieved using a 4 kHz LC low-pass filter with a 50 Ω shunt resistor connected in series with a 100 nF capacitor. The filter is matched to 50 Ω at the frequencies of interest and the shunt resistor gives a proper match at frequencies above this range.

10.1.5 Front end

The 'front end' of the transmitter refers to the RF frequency electronics that would be used to connect the remote unit to a power amplifier (PA, for transmitting) or a low noise amplifier (LNA, for receiving). It was decided that designing the PA/LNA was outside of the scope of the project as the design of such devices is already well covered in the literature, and the remote unit PCB could easily be connected directly to the input or output of a USRP without requiring these devices.

The front end implemented on the PCB includes two mixers and one power splitter/combiner. These parts were all sourced from Minicircuits, the TUF-3+ devices were chosen for the mixers and the PSC-2-2+ device is the power splitter/combiner. These devices are recommended by minicircuits for use at the HF/VHF bands and are designed to operate at low signal power (+7 dBm). The mixers are connected to the transmit/receive switch and the power splitter/combiner and serve the dual purpose of mixing the signals from the DACs to RF and the received signals to audio frequencies prior to the receive chain amplifiers. The transmit/receive switch is a Vishay DG455 low-resistance CMOS switch which uses two control lines from the DSP controller to switch the PCB mode between transmit and receive. On the other side of the mixers, the power splitter/combiner has the task of combining the two channels from the transmit chain together or splitting received signals prior to being mixed down by multiplication with the complex local oscillator signals.

10.2 Remote unit antenna

The remote will be connected to a poor quality antenna, as described in chapter 4.2 and summarised at the beginning of this chapter. The term 'poor quality' from the specification likely refers to an antenna that is opportunistically placed, possibly electrically short and of a poor match. The antenna is unlikely to be well constructed or raised a suitable distance from the ground. This will have implications for the design

of the remote system.

A poor quality antenna may not be an efficient radiator of energy, and there will be little control over the direction in which the energy will be radiated. The antenna might give a poor match to the transmit electronics which would have to be compensated for in the design of the transmit electronics so as not to cause damage to the power amplifier.

10.2.1 Buried antennas

A buried antenna has been investigated as a model for the poor quality antenna intended for use with the remote system. This type of antenna is assumed to be similar to the type of opportunistically placed antenna that may be attached to this antenna in real world applications. Locations such as national parks often have regulations in place prohibiting structures such as antennas to be erected above ground, in end-user cases such as these a buried antenna may indeed be the antenna connected to the unit. Chapter 9 investigates the performance of buried antennas with and without radomes.

10.2.2 Antenna matching

A poor quality antenna may be a very poor match to the front end RF electronics placed on the PCB board design presented here. This will cause reduced power to be radiated from the antenna and instead to be reflected back into the amplifier, possibly causing damage to the power amplifier components. To compensate for the differences in impedance matching electronics are required between the output of the remote unit PCB and the antenna [90]. Antenna matching techniques are already well known and antenna tuning units (ATUs) are widely available from companies like Rohde and Schwartz [110].

An ATU has the effect of transforming the characteristic impedance of an antenna at a given frequency to the same impedance of the transmitter amplifier (normally 50 Ω). This reduces matching losses as described in section 9.1.3, recovering up to 3-4 dB for poorly matched antennas.

10.3 Circuit schematics

The figures on the following pages give the complete circuit schematics for the remote unit. Figure 10.6 shows the microcontroller, digital to analogue convertor (DAC) and transmit path, serial interface and level shifters for enabling communication between the 5 V microcontroller pins and the 1.8 V DDS pins. Figure 10.7 shows the local oscillator, based around two DDSs, with the filter circuitry. Figure 10.8 shows the receive path, mixers and transmit/receive switch. Figure 10.9 shows the voltage regulators used to generate the various digital and analogue voltage levels required by the board.

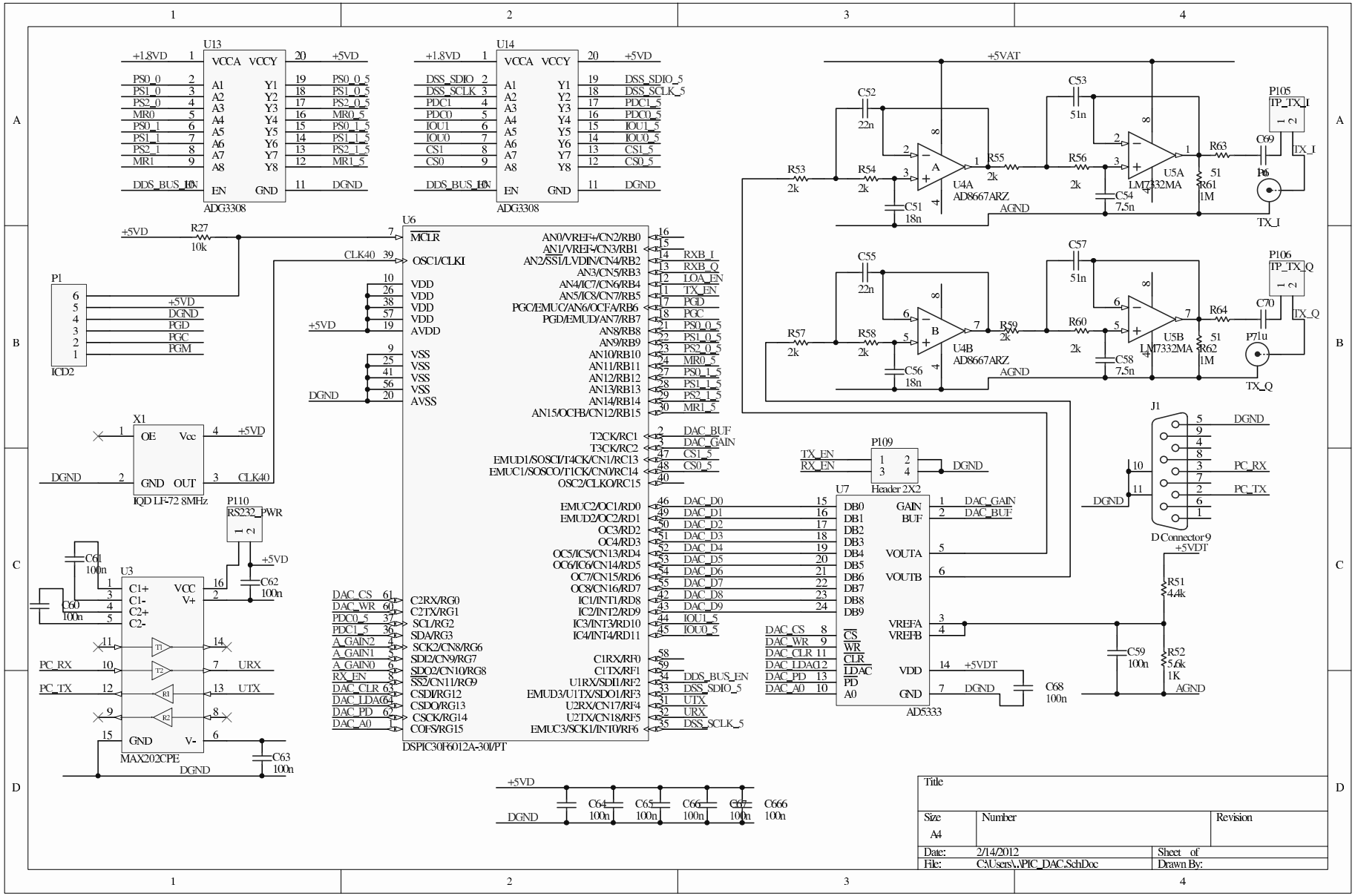


Figure 10.6: Schematic for the microcontroller, buffering, DAC and transmit path

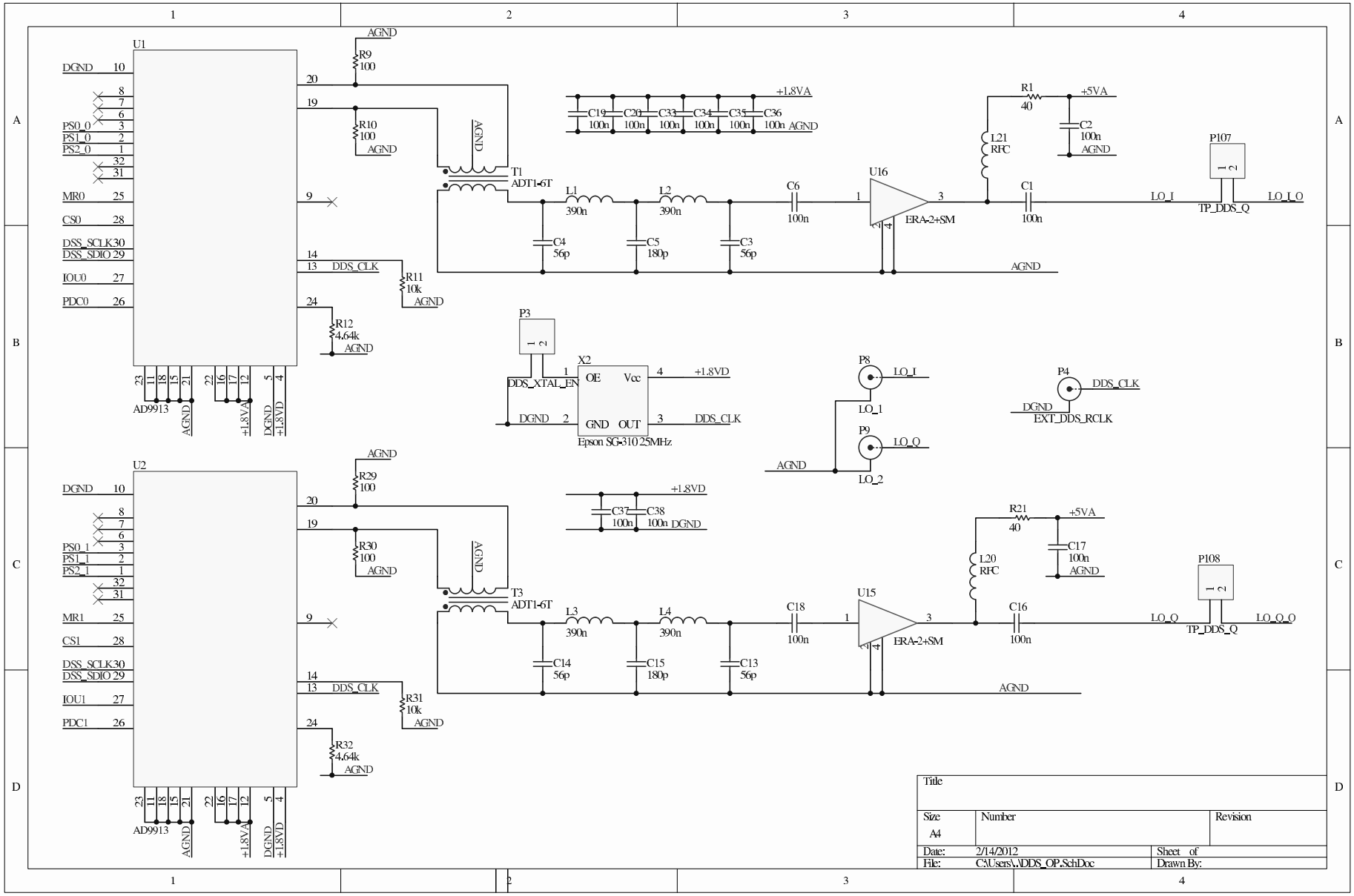


Figure 10.7: Schematic for the DDS based local oscillator

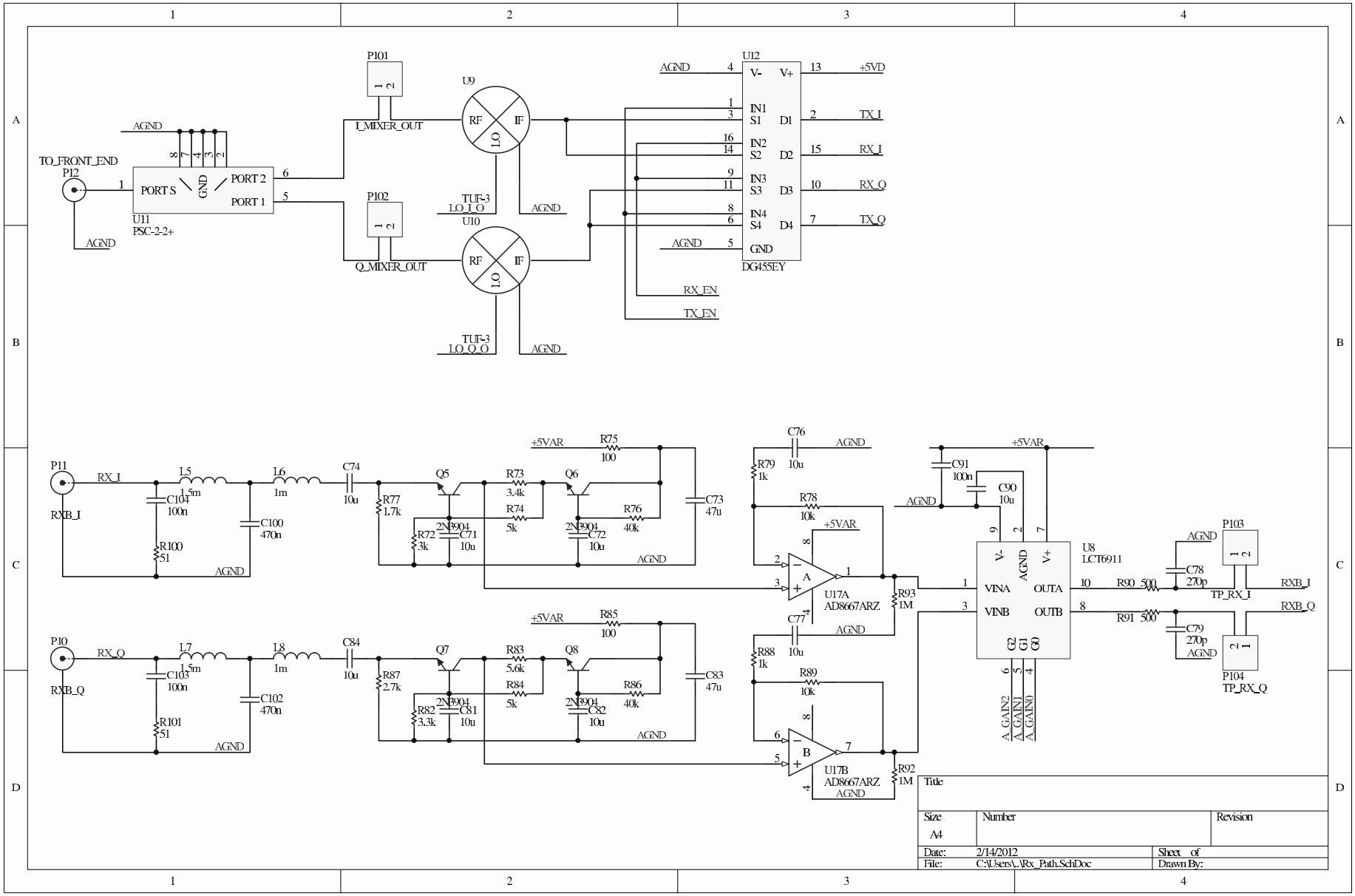


Figure 10.8: Schematic for the receive path and mixers

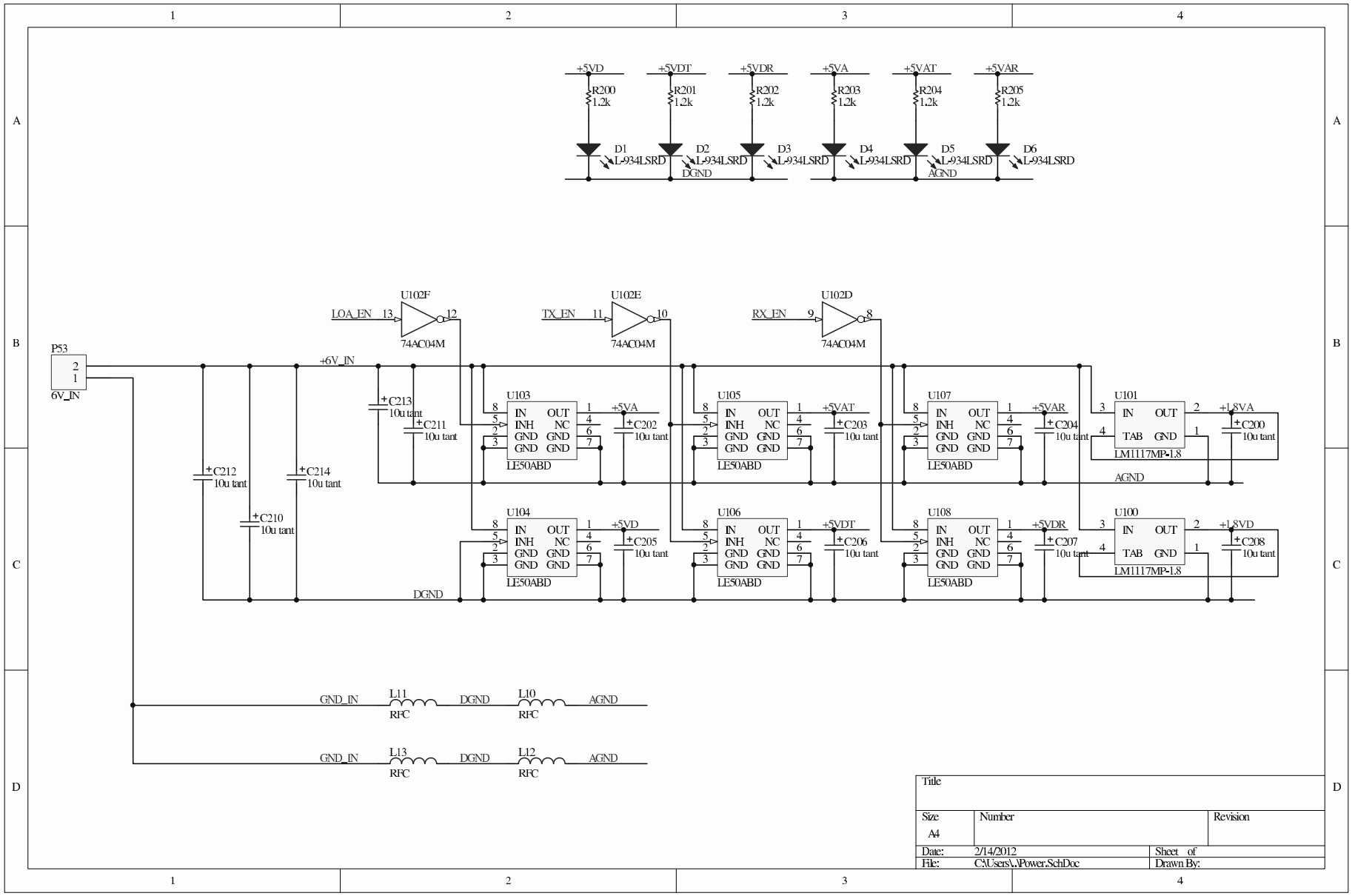


Figure 10.9: Schematic for the power supplies

10.4 Software

This section describes the software (or firmware) which will run on the microcontroller in the remote unit. As mentioned in section 10.1.1 the microcontroller selected for use with the remote unit is a Microchip dsPIC30F6012A digital signal processor. This is a low power device and considerably simpler and slower than some of the alternative offerings from companies such as Texas Instruments and Analog Devices. This processor has limited memory and computational resources compared to a faster DSP or FPGA (but much lower power requirements), which means that careful consideration must go into the selection and implementation of software algorithms for it.

10.4.1 Software overview

The software embedded on the board is required to perform a number of functions. It must initialise and control the microcontroller and board resources and implement the transmit and receive chains. At the time of writing only the transmit chain has been implemented, the receive chain software is still required to be written. This will involve writing software to sample the incoming signal using two ADCs and implementing a matched filter bank for the alphabet of command symbols. The transmit chain on the remote unit forms part of the primary communications link and is therefore of more academic interest to the receive chain. It is expected that the receive chain will be best implemented as a set of matched filters which will detect particular signals sent back from the base station, such as chirps, which will represent simple commands and repeat requests.

The software for the remote unit was written in Microchip C30, which is a language very closely based around C, specifically for Microchip's 16-bit microcontrollers and DSPs [111]. At this point the functionality implemented in the source code includes;

- **Forward error correction (FEC) encoding**, which encodes source data using a 1/2 rate QC-LDPC code with a codeword length of 2044 bits.
- **Packetisation**, which adds a header for transmission prior to sending a codeword.

- **Data modulation**, which modulates the codeword data using either binary- or quadrature- phase shift keying.
- **Output filtering**, which interpolates the output data and applies a root raised-cosine (RRC) finite impulse response (FIR) filter to the result.
- **Transmit scheduler** driven by interrupts, which pulls samples from the filter buffer and sends them to the DAC.
- **DDS configuration**, which conFigures the DDS local oscillators to generate a sine wave of a given frequency.
- **DAC configuration and arbitration**, which conFigures the DAC and provides an interface for sending samples to it.
- **Power rail control**, which provides an interface for activating and deactivating the power lines.

The functionality still required to be written includes;

- **Receiver chain**, which will be required to read samples from the DSP's on-board ADCs at regular, scheduled intervals and implement a matched filter on the acquired data stream.
- **Repeat packet generator**, which generates repeat packets for the various HARQ techniques discussed in section 7.7.
- **PC Interface**, which will use the universal serial asynchronous receiver/transmitter (USART) to read commands and a data stream to be transmitted in from a PC or other external device.

The software is available in appendix D.

Figure 10.10 shows a block diagram of the software required for the modem. Blocks with dotted lines indicate parts of the software which are not yet implemented. The following sections will detail the implementation of these techniques on the board.

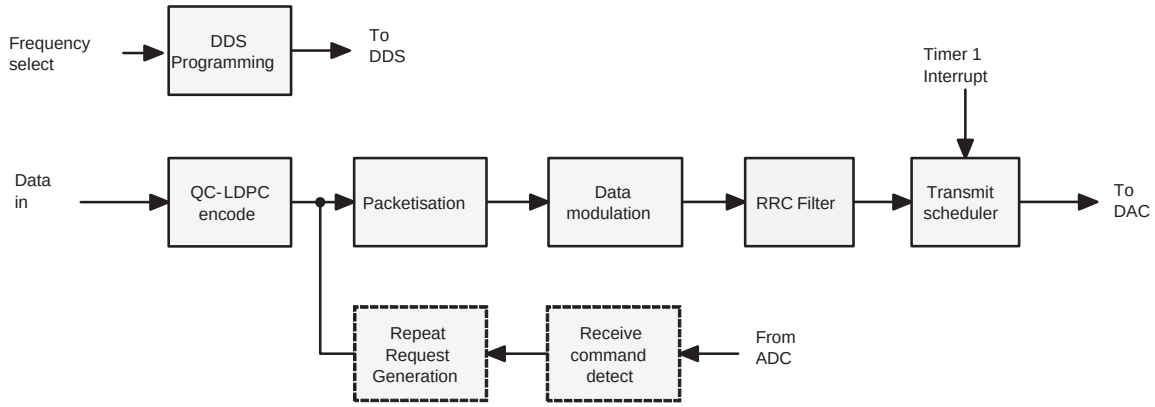


Figure 10.10: Block diagram of the remote unit transmit chain

10.4.2 The 'fractional' data type

At this point it is useful to briefly discuss the main data type used for representing analogue sample values within the DSP. The *fractional* data type is a 16-bit fixed point type suitable for representing data values in the range $-1 < x < 1$ [111]. This data type is defined from the standard 16-bit integer type, and is defined to have 1 sign bit and 15 fractional bits. The fractional type uses a twos-complement style representation for negative numbers, therefore $0x8000 = -1$ is the lowest representable number and $0x7fff \approx 1$ (or, more precisely, $0x7fff = (2^{15} - 1)/2^{15}$) is the highest representable number. To convert this data type into floating point representation, say for Matlab, an easy conversion is $y = ((x^{0x8000})/0xffff) - 0x7fff$, where x is the fractional type and y is the floating point type.

Microchip's in-built FIR libraries use the fractional data type and thus this type has been used for representing analogue data throughout the source code for the remote unit.

10.4.3 The transmit scheduler

The transmit scheduler consists of an interrupt service routine tied to timer 1 in the DSP. Additionally a first in first out (FIFO) buffer has been implemented from which to store samples from the FIR filter. The samples in the FIFO buffer are stored as a data structure of two *fractional* variables, representing the in-phase and quadrature components of the sample to be transmitted. Samples are pulled from this buffer every time the interrupt service routine executes and written to the DAC. A note of

the number of samples left in the buffer is kept and when this drops below a predefined safety margin, a flag is set that requests that the buffer be refilled. As the FIFO buffer is emptying, the code which runs the FIR filters is filtering a new buffer which will be copied into the FIFO after the number of samples within it is detected to be below the safety margin.

Refilling the FIFO buffer is time consuming in comparison to the time available between interrupts. It is for this reason that the safety margin on the FIFO buffer has been implemented. This is also the reason that the ISR sets a flag which tells the main program to refill the FIFO, as opposed to refilling it within the ISR itself. Adding such a lengthy operation into the ISR itself will make its execution time unacceptably long, to the extent that another interrupt will be hit before execution of the first one completes.

The transmit scheduler sets the data symbol rate of the transmit chain. It pulls samples from the forward end of the chain and as such everything behind it is synchronised to it. The sampling rate of the scheduler has been set provisionally at 10 kHz (this value gives a 10:1 interpolation ratio at the FIR filters for when using a symbol rate of 1 kHz). If it was desired that the symbol rate of the output data stream were to be increased, say to 4 kHz, then this would be easily achieved by increasing the sampling rate of the scheduler, which would mean that no further changes to the FIR filter code were necessary.

10.4.4 The FIR filters

There are two FIR filters which provide root-raised cosine (RRC) filtering to the in-phase and quadrature components of the transmit chain. These filters have been implemented using the in-built FIR functions provided by the Microchip DSP library [101]. This library does not provide support for complex types or complex filtering, so a complex RRC filter has been approximated by using two real RRC filters, one for each channel. These filters each maintain an output buffer for storing the results of the interpolation and filtering process of the modulated symbol stream.

Every time that the FIFO buffer in the transmit scheduler is refilled, the entirety of both FIR output buffers are copied into the FIR. Once this process is finished, the FIR filters are ordered to refill their output buffers in entirety, which happens while the transmit scheduler FIFO buffer empties again.

The FIR filters read data from two memory buffers at the output of the data modulators, one each for the in-phase and quadrature components of the signal. Once every sample of the modulator output buffers have been fed into the filters, the filter code requests that these be refreshed (see below).

The taps for the FIR filters were generated in Matlab, converted to *fractional* type notation and hard coded into the C code of the filter. One caveat of the Microchip DSP library is that for the in-built and heavily optimised FIR interpolate and filter function, (*FIRInterpolate()*), the number of taps has to be an integer multiple of the interpolation rate. As a root-raised cosine filter generally has an odd number of taps this poses a problem. To work around this a RRC filter was designed in Matlab to have 61 taps and the last tap was removed to give 60 in total, which is an integer multiple of the 10:1 interpolation rate. Using a 60 tap filter gives a suitable 6 symbol width to the RRC filter.

10.4.5 Data modulation

The data modulator is quite simple in operation. It takes packed data from an input data stream and assigns static values to the output buffer representing the modulated data. In this case the term ‘packed data’ refers to data sent in 16-bit words, where each word represents 16 bits of data (as opposed to one word per bit of data). The fact that it uses packed data is important to keep the memory consumption of the FEC encoder small enough to fit in the data memory of the DSP. This is discussed further below in section 10.4.6.1.

The modulator is capable of modulating data in either binary- or quadrature- phase shift keying (BPSK or QPSK) formats. To modulate the data, bits of each word are read into the modulator sequentially. Where a zero is detected, which corresponds to a symbol value of 1, the value 0x7fff is placed on the output buffer. Where a one is detected, corresponding to a symbol value of -1, the value 0x8000 is placed on the output buffer.

The output of the data modulator is double buffered. Two buffers are maintained and when the FIR filter requests a refresh of the modulator output buffer the output buffers are flipped, and the modulator outputs to the inactive one. The reason for this is to prevent the associated delay from modulating a new output buffer from impacting the upstream sections of the transmit chain. A circular buffer or similar

with a built in safety margin could have been used instead.

10.4.6 FEC Coding

The FEC coding technique chosen is LDPC, which allows the receiver to perform the blind LDPC turbo decoding which was evaluated over the course of this project. As described in section 7.5.3 encoding an LDPC code requires a large, non-sparse matrix multiplication (the matrices for a typical LDPC code have dimensional sizes in the order of hundreds or thousands). This is problematic for two reasons. First, multiplying a non sparse matrix with a vector can take a prohibitively large number of processor cycles (in the order of tens of millions). This is highly undesirable, especially when using a low power DSP running at a clock speed in the range of tens of megahertz with no instruction level optimisations for multiplications. Secondly, the amount of memory required to store the large matrix in is also typically in the order of megabytes, much larger than the limited memory of a typical low power DSP. The amount of data memory available on the dsPIC30f6012A is approximately 6kB, while the memory required to store a 2044-bit LDPC code (of the length selected for this project, please see below) is approximately 2MB. Clearly some alternative technique has to be used to generate the LDPC codes at the remote unit. Luckily, a family of LDPC codes known as quasi-cyclic codes has been discovered which can be encoded using a far less demanding algorithm [62][64][63]. These techniques are discussed further in sections 7.5.2 and 7.5.3. The next section will detail the implementation of such an encoder on the microcontroller.

10.4.6.1 Encoding QC-LDPC codewords

This section details the implementation of QC-LDPC encoding on the DSP controller. For an in depth discussion of this technique refer to [62] and sections 7.5.2 and 7.5.3.

The QC-LDPC code generated for use with the modem was a 1/2 rate code with codeword length of 2044 bits. The code was generated in Matlab (the source code of the code generation algorithm is available in the appendix) and hard coded into the remote unit's code. To create the code, all the possible lines through a point in a 3-dimensional finite Euclidean geometry with a dimensional length of $2^3 = 8$ are found. This results in a set of 73 lines each of length 8. A search on this set is then

conducted to find the different cyclic classes (i.e. those that aren't rotated versions of others in the set), and 9 such classes are found. Each of these cyclic classes are representative of the first row of a 511×511 sparse circulant matrix of row (and column) weight 8. The first two of these circulant matrices are taken and placed together to form a 511×1022 matrix of row weight 16 and column weight 8. Row and column decomposition is then performed on the constituent circulant matrices of this matrix to form a new 1022×2044 matrix made up of eight constituent 511×511 circulant matrices, each of row and column weight 2. The new matrix has a row weight of 8 and a column weight of four and is the parity check matrix for the QC-LDPC code.

After defining the parity check matrix, the next step is to find the generator matrix, which is the null-space of the parity check matrix. This consists of one 1022×1022 identity matrix and a 1022×1022 non-sparse parity bit generator matrix (more precisely the full generator matrix would have of an extra two rows due to the singularity of the parity check matrix, the rank of which is actually 1020, but these have been ignored for the purposes of slightly simplifying the code). The non sparse section of the parity check matrix is itself quasi-cyclic, and consists of four 511×511 circulant matrices. As only the first row is needed to represent each of these matrices, this matrix can be represented using approximately 256 bytes of information, representing a significant decrease in required memory resources when compared to a typical random LDPC code (which would require 511 times as much memory). These 256 bytes are defined in the source code as four arrays of 32 16-bit integers.

To encode the message on the DSP controller, a simple algorithm is used which is described in [62]. Two 511-bit feedback shift registers (FSRs) are implemented on the DSP controller and initialised to zero. For each of the four arrays imported from Matlab, 511 bits of the 1022-bit long message vector are sequentially checked and upon detection of a one, the array is added to the corresponding FSR. For every time a message bit is checked the FSR is rotated one bit. The FSRs are rotated in-place, there is no circular aspect to them, and every time a rotate takes place every bit in the register will move left one bit in the memory.

10.5 Interfacing with single board computers

A class of device currently gaining popularity is the *single board computer* (SBC) which implements a small, low powered computing system with an operating system and a large range of I/O options. These devices often use DSP or embedded media processors with peripherals including USART, ethernet, universal serial bus (USB), serial peripheral interface (SPI), video graphics array (VGA) out, and more. Linux is typically used as the operating system, due to its open source nature it is well suited to engineering development.

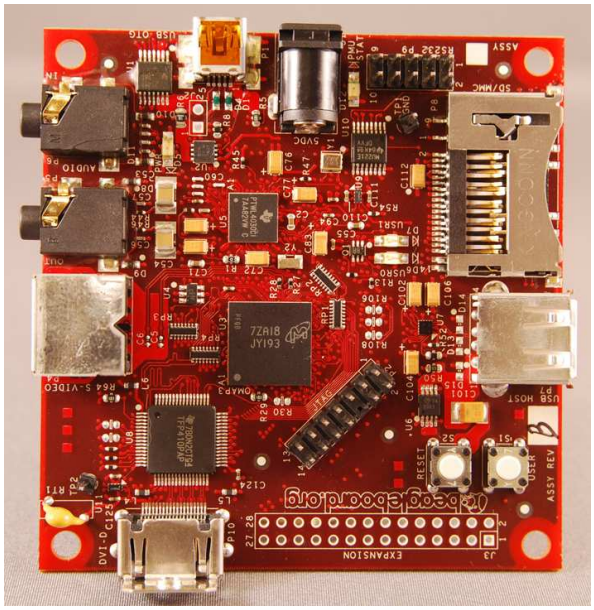


Figure 10.11: The Texas Instruments Beagleboard SBC

The Beagleboard is a popular, low cost SBC platform, developed by Texas instruments [112]. Figure 10.11 shows a photograph of this device. As of revision C4 of the Beagleboard, it is based around a TI OMAP3530 processor, which includes an ARM Cortex-A8 CPU and a TI TMS320C64x based DSP co-processor. On board there is stereo audio input and output, 256 MB of Flash memory, 256 MB of RAM, slave and host USB interfaces, an RS232 serial interface, JTAG interface and DVI HD-capable video connection. The board typically runs on Linux, with the Angstrom distribution, which is optimised for embedded systems.

There is a build of GNU Radio available for the Beagleboard. It is relatively simple to install and has been installed on a Beagleboard in the lab. The simple dial tone demo which comes with GNU Radio has been verified to be working, generating an output signal from the stereo audio output. However, no further work integrating the

existing GNU Radio transmitter code has been installed or tested on the device due to time constraints.

The speed and low power of the Beagleboard, coupled with the ease of implementation of the remote unit's algorithms, which have already been written to simulate the techniques in GNU Radio, makes the device a compelling platform upon which to base the remote unit. It should certainly be considered as an option if any further work is to take place to build upon this project.

10.6 Summary of transmitter unit PCB

The remote unit PCB, in its current state, has been found to consume approximately 210 mA at 5V while transmitting. This Figure was tested with a core processor clock of 64 MHz, at which speed a 2044-bit QC-LDPC codeword with an approximately equal number of 0- and 1-bits took approximately 312 ms to encode. The QC-LDPC algorithm, as implemented above, fitted comfortably in the processor's 6 kB of on-board data memory.

10.7 Further work

The remote unit board has been manufactured as a 'proof of concept', that it is possible to implement a very low power HF modem, capable of generating long QC-LDPC packets sufficiently quickly, modulating the data using BPSK or QPSK, upsampling and filtering the stream of modulated symbols and up-mixing to an RF frequency.

Unfortunately due to time constraints the receive chain on the PCB was never tested. As this design is from [109] it is expected that there will be few problems with the design. However some code will be required to be written to fully implement this feature. Further to this, a full evaluation of the performance of the receive chain should be carried out, including assessing the fitness of the DSP controller's on board ADC for use in the system. The missing part of the remote unit, in its current state, is the RF front end electronics that would connect the output of the PCB to the antenna for both the transmit and receive chains. There are two devices

required to be implemented for this, a low noise amplifier to amplify received signals from the antenna before they are fed into the receive chain of the PCB, and the power amplifier which will amplify the signal from the transmit chain to a power level sufficient to establish a link to the base station, at a BER of at least 1 kbps, when the signal is radiated through the poor quality transmit antenna connected to the remote unit. If the remote unit was reconFigured to use a constant modulus waveform such as offset-QPSK (OQPSK), the possibility presents itself of using a very efficient, non-linear amplifier such as a class-D type. These possibilities ought to be examined as potential implementation techniques for such a remote unit. To fully integrate such a change with the rest of the techniques being used in the system, the base station algorithms, in particular the blind turbo equalisation, would also have to be adapted.

The remote unit has not yet been fully tested, to the extent that actual data has been transmitted from at and been successfully received at a receiver. This was due to the limited time constraints of the project and effort being focussed on the LDPC turbo equalisation and developing the transmit chain firmware. Any further work should certainly assess the actual performance of the unit on a real or simulated link. Further, this device should be tested using batteries, to assess the length of time that it is possible for it to stay on air. Obviously this process should be performed when the power amplifier design has been finalised to give an overview of the capabilities of the device.

Chapter 11

Implementation of the base station

This chapter discusses the implementation of the base station in the project. A detailed account is given of how the software algorithms described in chapters 7 and 8 have been implemented within the constraints of the software defined radio (SDR) system chosen to implement the device. For details on the performance evaluation of these techniques, including simulation results and analyses, are presented in chapter 12.

11.1 Base station description

The *base station* is the device situated within a permanent structure, which receives data transmissions from the remote unit and transmits instructions and repeat requests to it. It is connected to mains power and a good quality directional antenna. The main requirement of the base station, as discussed in section 4.3, is a computational capacity high enough to implement the techniques discussed in chapters 6, 7 and 8. The complexity of the techniques presented preclude the use of most embedded solutions, and while it may have been possible to implement some of the techniques on a high speed FPGA, the expense in terms of money and required time of this approach was unfeasible given the constraints of the project.

For these reasons it was decided that software defined radio (SDR), discussed in the next section, would be used to implement the base station modem. In particular the GNU Radio platform was used, which is an open source SDR development platform (discussed below). Using SDR techniques allows the modem to take advantage of

the computational power of a PC, which even at entry-level specification, provides far greater processing power at a lower cost compared to embedded systems. Modern, multi-core PCs are well equipped to perform digital signal processing; filters can be implemented using simgle-input multiple-output (SIMD) processor instructions, allowing even very long, narrow bandwidth FIR filters to be executed at high speeds. GNU Radio signal processing blocks, which are written in C++, can be programmed to utilise the full power of modern hardware, allowing very complicated algorithms to be incorporated into a functional modem. The benefits and applications of GNU Radio to the laboratory and this project are further discussed in a conference publication [113].

11.2 Software defined Radio

In recent years, as the computational capacity of processing hardware has continued to grow exponentially, it has become feasible to implement radio systems fully in the digital domain, with digital hardware connected, via an analogue to digital interface, directly to an antenna. The term *'Software defined radio'* is defined by the P1901.1 standard of the IEEE; *"A radio is considered to be a software defined radio if some or all of the baseband or RF signal processing is accomplished through the use of digital signal processing software and can be modified post manufacturing."* [114].

The software can be resident on a personal computer or implemented as firmware on an embedded system. As SDR becomes more popular a number of companies have started offering SDR radio systems. Texas Instruments develop systems based on their digital signal processors [115], Rohde and Schwarz [116], Aerostream Communications [117] and Lyrtech [118] offer products built around third party DSP devices. SDR platforms developed for execution on a PC include an implementation of the well known Digital Radio Mondiale (DRM) specification, WinDRM [119], Sora [120], which is a system being developed in partnership by Microsoft and Beijing Jiaotong University, FlexRadio Systems' [121] products and GNU Radio [32] (discussed in the next section).

Additional hardware is required to interface between a PC and an RF output stage. Products available, at the time of writing, to do this include the SoftRock radio kit [122] and proprietary hardware for GNU Radio and FlexRadio's products. The SoftRock kits are a range of simple analogue mixer devices which each operate at

a fixed frequency defined by a crystal, various models are available for receive only and half duplex communications and GNU Radio can be used to generate the base-band signals to feed to this device. GNU Radio and FlexRadio Systems' platforms have associated proprietary hardware.

Small form-factor SDR (SFF-SDR) is a technology in which SDR systems are designed to be small in size and low-powered. This makes them feasible for use as battery operated devices. Notably, there is a GNU Radio build available for the Beagle Board [112], which is a single board computer (SBC) based around a Texas Instruments OMAP (Open Multimedia Applications Platform) processor, which unites an ARM9 core and a DSP core in one package, with a total power consumption of approximately 2W. The low power consumption of this device makes it appropriate for consideration as a candidate platform for the remote unit. Please see section 10.5 for further details.

11.3 GNU Radio

GNU Radio is the SDR system which was selected as the platform upon which the base station was to be implemented. It is an open source system which made it attractive for use compared to the other, commercial systems on offer. For clarity, at this point it would be useful to briefly describe some of the functionality of GNU Radio in terms of its application to the techniques described in this chapter.

Figure 11.1 shows a block diagram of an example SDR system defined in GNU Radio. The radio functionality itself is defined by a flow graph which connects signal processing elements, typically written in C++, together using Python scripting. The use of the Python scripting language allows many other third party APIs to be used to assist with graphing data and performing mathematical analyses, for example.

11.3.1 GNU Radio architecture

A radio is defined using GNU Radio by constructing a *flowgraph*, which defines links between *signal processing blocks*, which perform digital signal processing operations on data passing through them. The flowgraph is defined in the Python scripting language [123] and the signal processing blocks are written using a C++

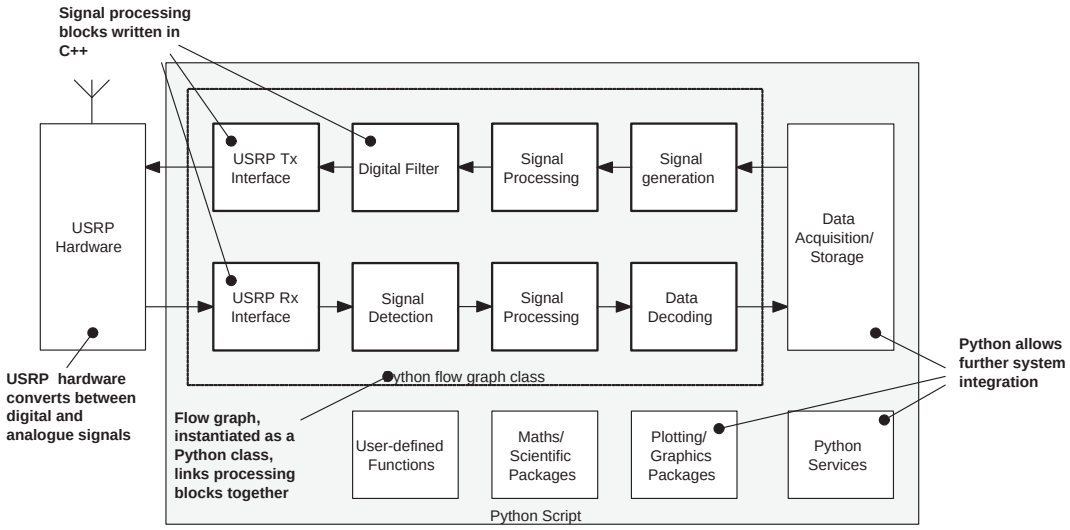


Figure 11.1: Block diagram of an example GNU Radio system

API included with the GNU Radio package. Communication between Python and the C++ code is achieved using the *simplified wrapper interface generator* (SWIG) package. Graphical user interfaces (GUIs) can be constructed with help from the integrated support for the wxWidgets GUI API.

The signal processing blocks are defined as C++ classes and have a hierarchical selection of types and inherited sub-types. The *top block* is the main block in a flow graph. This block is used to define connections between *hierarchical blocks*, which are sub-blocks instantiated as child classes of the main top block class. The top block class defines a number of functions used to connect together its child blocks and control the operation of the flow graph. The *connect()* function is used to define connections between the inputs and outputs of child blocks. The main control functions which will be described here are the *start()*, *stop()*, *run()* and *wait()* functions, which are all called from the Python script.

The *start()* function builds and runs the flow graph. This involves instantiating and initialising each hierarchical block where necessary and initialising the circular buffers which are used to transfer data between the running blocks (discussed in the next section). In addition this function verifies the flow graph, detecting errors and illegal patterns such as feedback loops within it. The flow graph will execute until a *stop()* function is called or an exit condition is detected from within the flow graph itself, say on completion of the transmission of a finite length message. The *stop()* function instructs the graph to exit immediately. The *wait()* function is a helper function for the python script which puts the script in a while-loop and returns when the flow graph completes. The *run()* command is a simple command which simply calls the *start()*

and *wait()* commands sequentially, starting the graph and resuming execution when the graph finishes.

The *stop()* command allows a flow graph to finish by itself and unloads the constituent hierarchical child blocks from memory. This then requires that the *start()* command initialise the hierarchical child blocks in the top block. This obviously forces them to restart in a newly initialised state. When the flow graph finishes itself, no blocks are reloaded from memory and any subsequent start command will restart all the blocks within the flow graph from the state that they finished in. Thus all memory buffers are internal to the blocks, and all state variables will be maintained. This is an important property of the flow graphs used to implement the automatic repeat request techniques described below in section 11.4.5. Unfortunately the circular buffers connecting blocks are cleared in either case which causes problems with the simulation and implementation of such techniques as described in section 11.5.3.1.

Each hierarchical block within the flow graph (henceforth simply referred to as a *signal processing block*), has a number of functions which are called from within the C++ code itself for the purposes of flow control. These will be briefly described here. A call to *set_output_multiple()*, normally called from within the initialisation code, will require the GNU Radio system to expect an number of output samples equal to an integer multiple of the argument passed to the function, which is itself an integer. Following from this, the *forecast()* function, which is overloaded as a function of the signal processing block, defines how many input samples are required to satisfy the requested number of outputs. It is possible to define synchronous block types which assume a 1:1 relationship between the inputs and outputs of a block which do not require this function. The *general_work()* function (named simply *work()* in synchronous blocks), is the overloaded function which defines the actual signal processing to be performed on the input stream. This function is required to copy the processed input to a given output buffer or buffers. The function is required (in the asynchronous type blocks) to call the *consume_each()* function, which tells the scheduler how many input samples were processed (this need not be exactly how many were requested in the *forecast()* function). The integer return value of the *general_work()* function defines the number of output samples copied to the output stream(s), or alternatively a -1 value can be returned to declare that this block has finished processing data. As of GNU Radio version 3.3, it is possible to define different return values (that is different numbers of samples copied) to each output stream. Refer to the source code for details [32].

When GNU Radio detects that the data flow through them has finished, due to a -1 value being returned by the *general_work()* function of a source block, any remaining data is fed through the flow graph and the flow graph exits. At this point any *wait()* functions running within the Python script will return.

11.3.1.1 The universal software radio peripheral

The universal software radio peripheral (USRP) and the its second version (USRP2), both manufactured by Ettus technologies [15], are devices used to interface software radio systems with the real world. Particular effort has gone into integrating this device with GNU Radio, and signal source and sink blocks have been defined to integrate interfaces to these devices easily into a flow graph.

The USRP and USRP2 are both systems that perform analogue to digital conversion (and vice versa), and digital up- and down- conversion of a digital signal to and from radio frequencies. The USRP uses a universal serial bus 2.0 (USB2) interface to a PC for data transfer which provides up to 16MHz of RF bandwidth. The RF front end is implemented by drop-in daughterboards available in a range of frequencies from DC to $>$5GHz. The device can support up to two transceivers (or two transmitters + two receivers) allowing up to 2×2 multiple-input multiple-output (MIMO) systems to be implemented. The USRP2 is a faster version of the device, with a maximum RF bandwidth of up to 25MHz using a gigabit Ethernet port to connect to the host PC and IQ signalling. The USRP2 is backwards compatible with the same daughterboards used on the USRP, but only one transceiver (or one receiver plus one transmitter) can be installed per unit. However it is possible to chain up to eight USRP2 devices together, allowing up to 8×8 MIMO systems to be implemented. See Figure 11.2 for a block diagram of the USRP.

The USRP has four 12-bit analogue to digital convertors (ADCs) which run at 64 MS/s for input and four 14-bit digital to analogue convertors (DACs) running at 128 MS/s [124]. The ADCs and DACs are provided by two AD9862 mixed-signal front end processors [125]. The spurious-free dynamic range (SFDR) of the ADC is 85 dB and the DAC is 83 dB [124]. The ADC has a theoretical SNR (from quantisation errors) of 74 dB (from the approximation SNR$= 6.02N + 1.76$, where N is the resolution of the ADC in bits [126]), and its measured SNR is 70.7 dB [125]. The SFDR is higher than this due to the processing gain of oversampling the input, the 85 dB Figure is assumed to have been taken from the datasheet [125], where the Figures have been

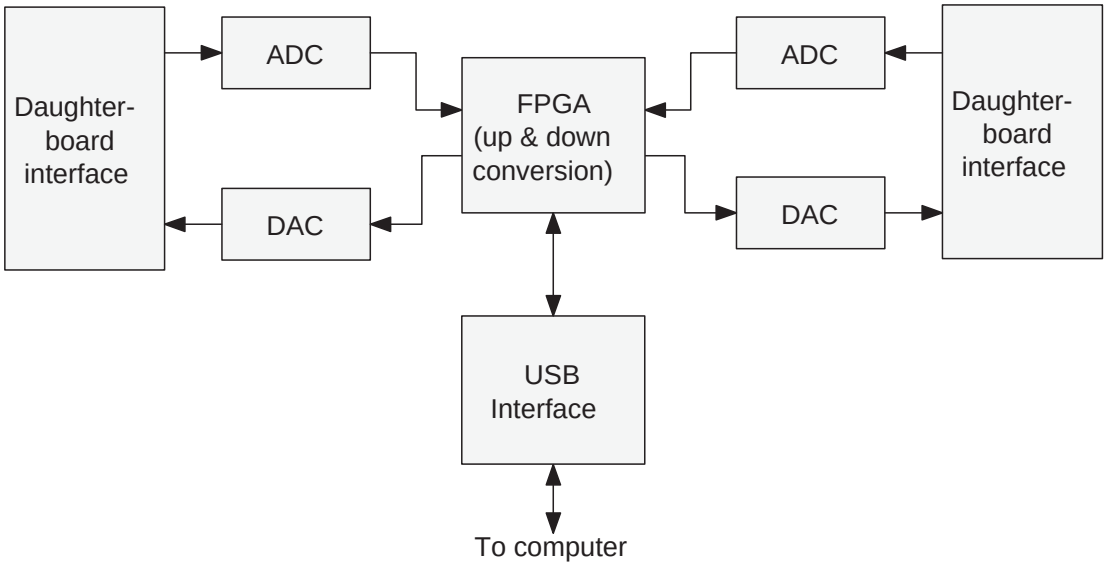


Figure 11.2: Block diagram of the USRP

measured for an input frequency of 5 MHz. As the input frequency increases (for HF band communications the carrier frequency is not down-mixed before being fed into the ADC), the SFDR will decrease. A doubling of the carrier frequency will result in a 3 dB reduction in SFDR. These metrics place constraints on the types of signals that can be received. The received signal will be difficult or impossible to decode if it is close to a signal (that is not pre-filtered out prior to being fed into the USRP) S dB stronger than the received signal where $S = R - T$, R is the SFDR of the ADC and T is the required SNR for equalisation and demodulation of the received signal.

The prices of the USRP and USRP2, at the time of writing, are 700 USD and 1400 USD respectively, putting them at the lower end of the price range for SDR systems. The daughterboards range from 75-275 USD each [15].

11.4 Base station software

As the base station was implemented as a software defined radio, the software for it was complicated and required to implement many functions that have traditionally been performed in the analogue domain, such as modulation and filtering. The base station software was built up from the GNU Radio SDR platform. Using this method simplified the implementation and simulation of the techniques used in the project. It also provided a large range of pre-written code for various commonly used techniques such as filters and MPSK modulation. Any other techniques required

(such as LDPC decoding and ARQ) were implemented as custom written signal processing blocks. It was possible to simulate the remote unit, channel and base station algorithms in GNU Radio in a single file, run it and send debug information to the screen or various files for later analysis. This process also exposed limitations within GNU Radio which are described in section 11.5.3.1.

11.4.1 Receiver algorithms

The receiver algorithms required in the system are shown in a block diagram given in Figure 11.3. From left to right these are a Costas loop which phase locks to the carrier of the received signal [100][10], a clock recovery block which detects and locks to the clock frequency of the received signal's data stream, a packet detector which detects receipt of a packet preamble and also generates an estimate of the state of the channel at the time that this preamble was detected, a semi-blind LDPC turbo equaliser which equalises and decodes the received packets simultaneously (described in sections 8.2, 8.2.1 and 8.3 for details on the theory of the operation of this block) and a repeat packet generator which generates ARQ repeat requests upon failure of the LDPC equaliser to correctly decode a block. The following sections detail the implementation of each of these blocks, describe problems with and suggest improvements to their current implementation.

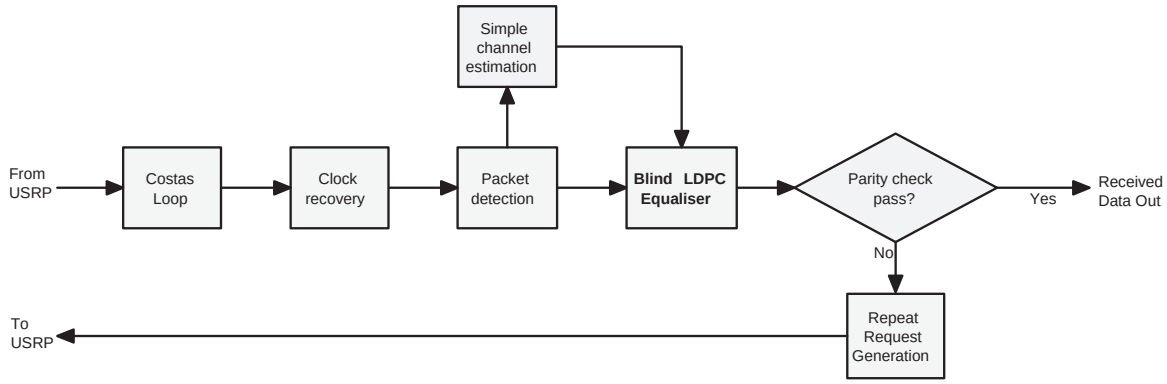


Figure 11.3: Overview of the software algorithms required in the base station.

11.4.2 Costas loop and clock recovery algorithm

These two algorithms are grouped in the same section as their code was sourced from the standard MPSK receiver block included in the current GNU Radio source

tree (version 3.3). This block takes the received baseband signal from the USRP, the Costas loop locks to the signal carrier (which will be close to 0Hz) and the clock recovery algorithm recovers the clock from the incoming signal and 'slices' it accordingly. This operation, known as 'data slicing', separates the individual symbols in a received signal from one another. The output signal from this block is a string of PSK symbols synchronised to the clock of the received signal. This block does not attempt to demodulate the signal. The operation and implementation of this block will not be further discussed here, for further details on the operation of this block, please examine its source code in appendix I.

11.4.3 Packet detection and channel estimation

The packet detection and channel estimation block accepts the clock recovered stream of PSK symbols from the MPSK receiver block and maintains a running buffer of the last m samples received, where $m = n + 2L - 1$, n representing the length of the preamble and L representing the length of the channel estimate, which should be set to be longer than the impulse response of the channel in terms of modulated symbols. The block operates by cross-correlating the n received symbols in the middle of the buffer with a known pseudo-random (PN) preamble sequence (which will be an M-sequence or similar). The magnitude of the result of this correlation is compared with a threshold value. If the result is higher than this value a preamble is assumed detected and resident within the block's buffer. The threshold value chosen must be sufficiently low that packets received with a low SNR (such as those received within a fade) can be detected, but high enough that the chance of detecting false positives due to noise is minimised. Adaptive techniques, for example based on SNR measurements, can be used to set the threshold to ensure that it is maintained at an optimal value.

Upon detection of a received preamble, the algorithm cross-correlates the preamble sequence with every contiguous block of n samples in the input buffer, the resultant values are then normalised by division by n. The result of this operation forms a vector of $2L - 1$ complex cross correlation results which represents the impulse response of the channel through which the preamble sequence passed prior to being received. From this vector a search is conducted to find the L contiguous elements with the greatest sum of absolute magnitudes, the resultant L-element vector is the channel estimate, h_e.

Once this process has completed, the GNU radio processing block used to implement this technique sends the channel estimate then the $c+L-1$ channel-corrupted LDPC codeword bits to the LDPC turbo equaliser. The variable c in this case refers to the codeword length of the LDPC code, and the extra $L-1$ bits are required by the equaliser as they correspond to the extra information 'smeared' by the channel.

As the last $L-1$ bits of the preamble may cause inter-symbol interference with the first $L-1$ bits of the LDPC codeword payload a short delay of symbols corresponding to zero-bits is left between the preamble and the payload. The reason for this is that, in its current implementation, the equaliser assumes that the LDPC code is preceded by zero bits, that is, the initial state of the channel is assumed to be zero. This will not be the case if there are still parts of the preamble section in the channel when the payload is first received. The equaliser can easily be changed to assume that the initial condition of the channel corresponds to the state that the last $L-1$ bits of the preamble would leave it in, but in the interests of being able to easily change the preamble sequence it was deemed easier to insert a delay. The delay is implemented in the code by simple adding some extra 0-bits to the end of the preamble sequence.

11.4.4 Semi-blind LDPC turbo equaliser

The semi-blind LDPC turbo equaliser is based on a technique described by Gunther et. al. [87]. The theory of this technique is described in sections 8.2, 8.2.1 and 8.3. This section will describe some of the issues encountered when implementing the technique in GNU Radio and assumes familiarity with the previously mentioned chapters and [87]. For clarity of exposition the block diagram of a semi-blind turbo equaliser is reproduced in Figure 11.4.

From Figure 11.4 it is evident that there are feedback paths from the LDPC decoder to the BCJR equaliser both through the extrinsic probability calculation and through the channel equaliser. As GNU Radio does not currently (as of version 3.3) allow feedback paths in flow graphs it is necessary to implement everything in the block diagram as a single signal processing block in C++. The block is required to equalise, demodulate and decode the incoming data symbols with prior knowledge of the LDPC code used to encode the modulated data stream at the transmitter and output a stream of message bits.

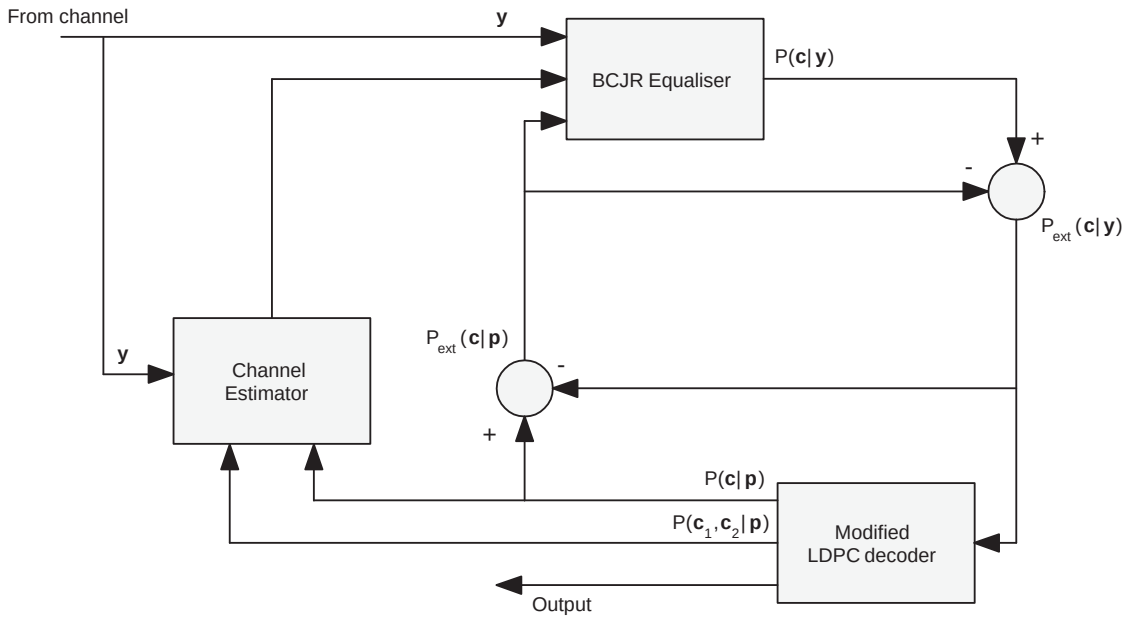


Figure 11.4: Block diagram of a semi-blind turbo equaliser

This algorithm is relatively complicated and as such the C++ code was split into multiple sections, each roughly implementing one block of Figure 11.4. The implementation of each of these blocks will be described in the following subsections.

11.4.4.1 The BCJR equaliser

The BCJR equaliser [82] is a maximum likelihood sequence estimation (MLSE) equaliser used to convert a sequence of received symbols into a sequence of posterior probabilities of codeword bits for entry into an LDPC decoder. It is described in section 8.2.1 and for a particularly good description of the implementation of the technique see [81].

The BCJR algorithm has been implemented in C++ for the project as described in [81], using linear probabilities contained in double data types for all the variables. On every step through the trellis the α and β probabilities are normalised by dividing with the sum of their magnitudes at that step. This ensures that the probabilities do not become so small that the granularity of the double data types used truncates them to zero, as (especially with longer sequences, which correspond to longer LDPC codeword lengths) this can become an issue rather quickly. The algorithm is fed with the received channel symbols, a channel estimate as given by the packet detector described above, an estimate of the noise and a vector of log-likelihood ra-

tios (LLRs) representative of the extrinsic likelihoods derived from the output of the LDPC decoder. The algorithm outputs updated LLRs of the codeword bits believed to have been modulated and passed through the channel to result in the sequence of symbols observed at the receiver. As the code is implemented using linear probabilities, it is necessary to convert the input LLRs to linear probabilities and convert the output probabilities back to LLRs.

The complexity of the BCJR algorithm increases geometrically with the length of the channel estimate used (L). The number of states required for each point in the trellis is equal to $2^L/2$. As such, using large values of L, such as those required for high data rates with fast fading channels, the required processing overhead could become an issue. In addition the number of symbols in the modulating constellation determines the number of state transitions from each node of the trellis. However, using a larger number of symbols will entail a shorter packet length for a given length codeword mitigating this somewhat. The implementation of the algorithm has been used successfully during the project with channel lengths of up to $L = 9$ and codeword lengths of up to $n = 2044$.

The BCJR algorithm has been adapted to use multiple channel estimates through the sequence to allow the LDPC semi-blind turbo equaliser to generate more than one estimate throughout the decoding process of a single codeword. For a fixed channel the channel estimate is kept the same for all the γ transition probabilities and is fed into the algorithm. To convert this to use a time-variant channel is a relatively trivial operation which simply requires that the channel estimate be changed at regular intervals throughout the generation of the γ transition probabilities. The γ transition probabilities are then worked out during each interval with a channel estimate generated for the corresponding time.

Ideally it is envisioned that the BCJR algorithm will be rewritten using logarithmic probabilities such as LLRs for the generated variables. This should reduce processing overhead due to the many multiply operations being replaced with computationally cheaper add operations. Also under-run errors associated with multiplying very small values together in floating point data types becomes less of an issue.

11.4.4.2 Generalised LDPC decoder

The standard LDPC decoder [61], which uses the belief propagation algorithm, is described in section 7.5.4. This is a relatively simple algorithm to implement in C++ in and of itself. However it requires a large number of multiply operations as the pseudo-posterior probabilities are generated. For that reason is was decided to implement the LDPC decoder in log space from a technique described by Leung et. al. [67][68]. This technique is described in more detail in section 7.5.5 and in practice reduces the execution time of the decoder considerably, especially for long codes. It is also suited to the LLR input from and output to the extrinsic probability calculations between the LDPC decoder and the BCJR equaliser.

The '*generalised*' LDPC decoder differs from a standard LDPC decoder in that it additionally generates pairwise pseudo-posterior probabilities for use with the channel estimator. The algorithm for generating these additional probabilities is detailed in [87]. The pairwise probabilities are the probabilities, $P(x_k, x_j|\mathbf{y})$, where c_k is the k^{th} codeword bit, c_j is the j^{th} codeword bit and $\mathbf{y}$ is the vector of n extrinsic probabilities derived from the output of the BCJR algorithm. The time required to evaluate every pairwise probability scales as the square of the codeword length and can be considerable. Additionally the memory requirement of an algorithm which is optimised to generate these probabilities fast is very large. The reason for this is that it is not desirable to represent any matrix which requires a large number of successive look-ups in any memory optimised sparse-matrix structure (such as Matlab's sparse type), as a binary lookup through a set of nonzero elements is required every time an index is checked. This reduces the access time considerably. Therefore it is better to hold the entire matrix in memory at the same time and use addressing to access the elements. When generating the pairwise probabilities for an $N \times M$ matrix it is necessary to hold a $N \times N \times M$ 3-dimensional matrix in memory as a reference. With large number, N, of 4-byte floats (such as $N = 2000$), this memory requirement can become huge ($2000 \times 2000 \times 1000 \times 4$ bytes ≈ 16 GB, for a 1/2 rate code of $N = 2000$) and well out of the range of a standard contemporary PC.

In practice, only those probabilities where $|i - j| < L$ are required, so only computing these reduces the complexity of the algorithm substantially. Also, if only these probabilities are required to be generated, the size requirement of the large 3-dimensional reference matrix can be reduced, using a simple remapping of the indices, to $N \times (2L - 1) \times M$. As $L << N$ the memory requirement of this matrix is greatly reduced. The current implementation of an $N = 2044$ LDPC semi-blind turbo

equaliser fits comfortably within the memory of a laptop with 2 GB of RAM.

Unlike the standard LDPC decoder, there is as of yet no known method of performing the pairwise calculations in log-space. Currently the probabilities are evaluated by converting the prerequisite log-space variables within the standard LDPC decoder to the linear domain and evaluating the pairwise probabilities from them. The problem stems from having four separate probabilities to represent (from the four combinations of a pair of binary variables) rather than the two probabilities embedded in a typical LLR. It is suspected that, given the discovery of a valid modification to the algorithm, moving computation of these probabilities to the log domain would both reduce the execution time of the technique by replacing multiply operations with faster addition operations, and more importantly reduce the sensitivity of the algorithm to arithmetic under-run errors that are observed occasionally in the linear domain. As protection against these under-run errors maximum and minimum acceptable values for the log-space variables are enforced before returning to the linear domain.

11.4.4.3 Channel estimator

The channel estimator within the semi-blind turbo LDPC equaliser is based on an adaptation of the Baum-Welch expectation minimisation (EM) algorithm [83][84]. Section 8.3 details the theory of operation of this part of the equaliser. This section details the implementation of the channel estimation algorithm and will use the same variable names as [87].

The vector $\mathbf{r}_n^{[p]}$ and the matrix $\mathbf{R}_n^{[p]}$ are evaluated from the posterior probabilities $P(x_k|\mathbf{y})$ and the pairwise probabilities $P(x_k, x_j|\mathbf{y})$ respectively, from the generalised LDPC decoder algorithm. It is necessary to provide limiting values to the algorithm while working these values out, which supply the algorithm with the values resident in the channel before the LDPC code is received. These values are required to state that probabilities of bits resident in the channel prior to receipt of the codeword are zero and are required when evaluating the prerequisite likelihoods for $\mathbf{r}_n^{[p]}$ (equation 8.5) and the prerequisite pairwise likelihoods for $\mathbf{R}_n^{[p]}$ (equation 8.6).

On every iteration it is required to solve $\mathbf{R}_n^{[p]}\mathbf{h}_n^{[p+1]} = \mathbf{r}_n^{[p]}$ for $\mathbf{h}_n^{[p+1]}$ (in fact this may be required to be solved multiple times per iteration, once per generated channel estimate; multiple channel estimates are described in the next paragraph). This

calculation is implemented in the channel estimator code using LU factorisation, the functions required to perform this factorisation are resident in the 'ublas' namespace of the C++ *Boost* libraries (*Boost* is a popular library of add on functions for C++ that is itself a prerequisite of GNU Radio). For more information on this operation refer to the source code of the LDPC turbo equaliser in appendix C.

It is possible to generate multiple channel estimates over the course of equalising a single received LDPC codeword. To perform this operation a neighbourhood is defined over the prerequisite likelihoods and pairwise likelihoods to evaluate $\mathbf{r}_n^{[p]}$ and $\mathbf{R}_n^{[p]}$. These are defined by the variables l_1 and l_2 in equations 8.4 and 8.5. While comprehensive testing of the best neighbourhoods has not been carried out, good results have been observed by defining neighbourhoods as follows: first split the received LDPC codeword into n_g groups of roughly equally sized sets of elements, each of length roughly N/n_g where N is the length of the LDPC codeword. Define n_g neighbourhoods of $3N/n_g$ elements where each neighbourhood comprises the n_g elements of its corresponding set plus the $2n_g$ elements of the adjacent sets. For the first and last sets of the codeword it is obvious that the neighbourhood size should be $2n_g$ as these sets only have one adjacent neighbour. For each of the n_g neighbourhoods $\mathbf{R}_n^{[p]}\mathbf{h}_n^{[p+1]} = \mathbf{r}_n^{[p]}$ should be solved for $\mathbf{h}_n^{[p+1]}$ to provide n_g separate channel estimates which can be fed back to the BCJR equaliser. The value of n_g should be set long enough that an accurate channel estimate can be found for each neighbourhood in noise, but short enough that the equaliser can adapt quickly enough to changes within the channel.

11.4.5 Repeat request techniques

This section discusses the implementation of various repeat request techniques under GNU Radio. For details on the theory of ARQ refer to section 7.7 and for details of the theory of a new ARQ technique developed as part of this project please refer to section 7.10.

Three repeat request techniques were implemented on the GNU Radio platform, a random repeat HARQ technique, reliability-based HARQ (RB-HARQ) [72] and a the new technique, random sum of elements HARQ (RSE-HARQ). As mentioned above these techniques are described in section 7.7, but they will be briefly summarised here.

11.4.5.1 Random repeats

For the random repeats technique, when a packet is received and decoded incorrectly at the receiver, a repeat request is sent back to the transmitter. The transmitter, upon receiving this request, prepares and sends a shorter packet containing a randomly chosen set of the constituent bits of the original codeword. This set will have bits chosen by a random number generator which is synchronised to an identical random number generator at the receiver. When the receiver receives this subsequent packet, it combines the repeated bits with the originally received codeword, generating an updated set of codeword log-likelihood ratios (LLRs) which are then fed into the FEC decoder. The effect of these subsequent repeats is that the received codeword SNR is increased incrementally, with receipt of each subsequent repeat packet. Eventually the set of received LLRs, which form the prior probabilities into the FEC decoder, are accurate enough that the original packet can be decoded error free.

11.4.5.2 Reliability-based HARQ

This technique works similarly to the random repeat method detailed above, except that the sets of bits used to construct the repeat packets are not randomly generated, but are chosen by the receiver based on the results of the FEC decoder [72]. When a packet is received and decoded incorrectly, the receiver examines the outputs of the FEC decoder, and finds the set of n bits that have the lowest posterior likelihoods, that is, those which it is least sure about. The receiver then constructs a repeat request to send back to the transmitter which embeds the locations of these lowest likelihood bits within it. This feedback path, however, requires a relatively good, error free channel, and is thus unsuitable for a HF link.

11.4.6 Random sum of elements HARQ

This technique again works in a similar way to the random repeat technique described above. The receiver, on receipt of an incorrectly decoded packet, sends a simple repeat request to the transmitter. The transmitter then constructs a packet with repeat information embedded within it. In contrast to the random repeat technique, however, each bit of this packet is determined by finding the linear combina-

tion of a randomly selected set of bits of the original codeword. The reasoning for this is that it is more likely that a bit of low likelihood will be retransmitted, which increases the combined SNR of the prior probabilities of the packet at the receiver. The packet is then input into a modified FEC decoder, separate to the rest of the codeword, as supplementary information. For more details on this technique, including the modifications required for a standard LDPC decoder, please see section 7.10. This technique is novel and was first presented in [127].

11.5 Implementation of ARQ techniques on GNU Radio

For each of the techniques shown above, the ARQ system has to generate repeat packets at the transmitter and recombine the information embedded within them with the originally received packet at the receiver. For random repeats and RB-HARQ, the repeats are combined with the originally received codeword before feeding into the FEC decoder and for the RSE-HARQ the repeat information is fed into the LDPC decoder together with the original information.

The repeat packet generators are implemented as signal processing blocks within GNU Radio. These blocks are designed to pass information straight through them, from the LDPC encoder to the modulator, when normal transmission is required, and to block further input from the LDPC encoder and generate repeat packets from the previously sent packet, which is temporarily stored in an internal buffer. The repeat packets are generated according to the techniques described above, with the random repeats and RSE-HARQ blocks requiring no further input (other than the knowledge a repeat has been requested), and the RB-HARQ blocks requiring a list of bits to resend.

At the receiver, blocks were created for combining the received repeat packets with the originally received packets. For RB-HARQ and the random repeat techniques, this was a relatively simple matter of keeping a record of the original packet and finding the means of those bits received as part of the repeat packets (as defined by a pseudo-random number generator for the random repeat technique and as instructed by the RB-HARQ technique). In terms of LLRs, which is how each bit is demodulated by the soft decoder, finding these means is performed by successively adding the received LLRs to the corresponding bits of the originally received packet.

A buffer of the received packet is kept within the repeat combiner block for these two techniques and updated internally with each successive repeat. This buffer is then sent to the LDPC decoder. For the RSE-HARQ technique, a separate block was not required to combine repeats prior to sending to the LDPC decoder. As this technique combines repeat packets with the original received packet at the decoder level, as described in section 7.10 both the original packet and any subsequent repeats are fed into a modified LDPC decoder.

The receiver's LDPC decoder required modifications for all the HARQ techniques. For the random repeat and RB-HARQ techniques, these are a simple matter of the implementation of a feature whereby any incorrectly decoded packets are not forwarded to the output of the block and a flag is raised that can be checked by code outside the block to prepare a repeat packet. For the RSE-HARQ technique, upon an incorrectly decoded packet, a flag is raised to instruct the outside code to send a repeat and the decoder itself switches modes to accept repeat packets as inputs and integrate them into the decoding process as described in section 7.10.

11.5.1 Reconfiguring the flow graph

The flow graph construct in GNU Radio is fairly difficult to reconFigure while it is executing. The manner of operation is to define a flow graph by instantiating, configuring and connecting blocks together, then execute the *run()* command from within the Python script, which performs validation checks on the flow graph and executes it. The flow graph executes in a separate thread (or threads) to the main script, exiting upon either a *stop()* command executed from the script, or an exit state detected from within the graph itself (for example when a finite length message has been entirely transmitted). The *run()*, *stop()* and related *wait()* commands are described briefly above in section 11.3.1 and defined in the GNU Radio documentation and source tree itself [32].

The nature of the HARQ techniques dictates, as described above, that the blocks have some form of memory for creating repeat packets and combining received repeat packets together with the originally received packet. This memory manifests generally as some sort of buffer plus state information within the block. The unpredictability of the HARQ techniques implies that the flow graph either needs some kind of intelligence and capacity to self-reconFigure or some means of being reconFigured from outside itself (presumably from the Python script from which it was

called). The former case is, in the current version of GNU Radio, unfeasible due to the illegality of feedback loops in a flow graph (at the Python level) and lack of suitable messaging interface for sending control messages from one block to another preclude the flow graph from being able to make any significant changes to itself. Also disconnecting and reconnecting signal processing blocks within a running flow graph is not possible. It is, however possible to reconFigure the graph from the Python script which called it.

Basic reconfiguration of the flow graph is possible when the graph is in a wait state (a wait state is defined here as the state a flow graph is left in after a *run()* command has executed within the Python script), without having had a *stop()* command issued to it. At this point all the blocks are resident in memory in the same state as they were left in as the last data passed through the flow graph. Here, the flow graph is still technically running even though no further data is being sent. A command can be issued to a source block to send more data and it will run through the flow graph similarly to any data the flow graph was initially set up to send. Additionally a command can be issued to a block downstream from the source block (such as a repeat packet generator) to send data in lieu of the source block, although this operation can cause problems with execution of the graph as described in the next paragraph. Unfortunately it is not possible, when in this state, to perform any *disconnect()* or *connect()* operations to reconFigure the pattern of the source blocks within the flow graph. For this the graph must be locked using the *lock()* and *unlock()* commands which will not be covered here (see the GNU Radio source code for further details), these commands have the same effect on the memory and state of the blocks within the flow graph as the *stop()* command so are of no use in the context of the implementation described here.

Another issue with the reconfiguration of a GNU Radio flow graph which was encountered during the implementation of the HARQ techniques was that the first block of the flow graph has to send data out. If no data is sent, and the block simply returns a -1 value (which causes the flow graph to exit), no further upstream processing will be attempted. This means that if a repeat packet is to be generated, where the return packet will originate from a block upstream from the first block in the flow graph, the first block must ‘fool’ the flow graph into thinking that it has sent some data. The subsequent -1 return value can then be generated by the upstream block (the repeat packet generator) when the repeat packet has been sent in its entirety. The first, source data block ‘fools’ the GNU Radio system in this way by simply returning a 0 when a repeat is being generated, instead of a -1. No data is copied by the block

onto the output buffer and the 0 return value informs the GNU Radio system of this fact. This is of no consequence as the second block in the flow graph, the repeat request block, does not require any input from the source block.

11.5.2 Implementing the HARQ simulations

This section describes how the HARQ techniques were simulated on the GNU Radio platform, in the context of what has been discussed in the previous sections. For further details on the setup and results of the simulations, please see section 12.5. For each of the three techniques tested, a separate script was written in which a relevant flow graph was defined. Only the forward path was simulated, the feedback path (through which the repeat requests travel from the receiver to the transmitter), was assumed to be perfect. In each script the entire system (including transmitter, AWGN channel and receiver) was implemented in one single flow graph.

To simulate the systems in operation, a simple algorithm was used which is shown in Figure 11.5. This Figure shows an algorithm by which it is possible to implement these techniques on GNU Radio. The configuration steps (higher in the diagram) are all operations performed by calling functions of the signal processing blocks themselves. These functions are programmed in C++ as member functions of the block and made available through the SWIG interface to the Python script. When configuration is complete the *run()* command is called from the script, and when this is finished the Python script checks whether the received, decoded codeword satisfies its parity check. If this is so, and the entire message has been sent, the algorithm exits. If the parity check message has passed but the message has not been sent in its entirety the flow graph is conFigured to send the next LDPC packet. If the parity check has not passed the flow graph is reconFigured to transmit a repeat packet through the channel. This process then repeats until the entire message has been sent.

It is important to note that the only error detection performed on the received and decoded codeword is the satisfaction of the parity check equation. It may be possible for the decoder to converge to an incorrect codeword, which passes the parity check equation but is still invalid. This will be discussed in the results in section 12.5.

11.5.3 Problems with the HARQ implementations

While the HARQ techniques have been simulated using a GNU Radio script some problems have been identified with the GNU Radio platform which makes the implementation of these techniques difficult to simulate as part of the main modem system. There are multiple reasons for this which involve both the fundamental architecture of GNU Radio and the integration of the new technique with the main modem system. These will be summarised in the following sections.

As the HF channel can be poor and continuously variable, with a high probability of bit errors without adequate coding and channel equalisation, there may be issues integrating HARQ techniques with the rest of the system. These issues stem from the fact that any repeated packets will have to be encoded with forward error correction redundancy themselves, so as to provide resistance to errors which could cause the repeat packets to further degrade the quality of the received packet. Ideally this would not be a major issue for any of the techniques, as in the random repeats and RB-HARQ cases, the repeat packets are combined with the original packet before decoding, therefore reducing the SNR of the received packet. In the case of RSE-HARQ, the fact that the received repeat packets are used integrally in the decoding process lends a similar resistance to noise as would be found from the originally received bits of the LDPC packet. Indeed, under AWGN, all three of these techniques have been simulated under these conditions with success (the results are provided in section 12.5).

The problem with implementing HARQ techniques in the system being built for the project stems from the fact that channel equalisation is performed as part of the FEC decoding process. This turbo equalisation method, described above in section 8.3, finds a channel estimate for a given message which has been encoded using an LDPC code with a known structure. If any packets are received without being encoded properly in a full LDPC packet of the structure expected by the receiver, the likelihood of correct decoding of the encoded message is very low. Further to this, the channel estimate generated by the LDPC turbo decoder can occasionally be observed to be completely incorrect. This may stem either from a received signal with low SNR or problems with the equaliser just failing to converge on the correct channel, occasionally, the equaliser can converge to a LDPC codeword that, while valid for the set of codewords, is completely incorrect. The symptom of such errors is a very high bit error rate. However these may prove to be difficult to detect in a practical system. This illuminates a need to perform some kind of SNR estimation on

the received signal to guide the decoding and an error checking code to be applied to the data before performing the LDPC encoding.

At the time of writing, there is no known way to integrate the new HARQ technique, which requires modifications to the LDPC decoder, with the semi-blind LDPC turbo equaliser. The extra degrees of freedom inherent in using the minimum distance of the LDPC codewords to equalise the channel requires that every bit can be provided a valid channel estimate. While it will be possible to provide a channel estimate for the LDPC code, further repeat packets, even if appended directly to the end of the codeword, will not have a valid estimate. It may, however, be possible to add a small number of repeat bits onto the end of a given codeword under the assumption that the channel will not change substantially from the final channel estimate given for the codeword (to provide extra redundancy, say, for an adaptive system).

11.5.3.1 Integration with GNU Radio

Due to the problems described in section 11.3.1, there have been some problems identified in the HARQ systems stemming from the fact that the circular buffers are reset every time the *run()* command is executed (that is, every time a new packet or repeat packet is sent, see Figure 11.5). In a more in depth simulation than those undertaken here, or possibly a real system, this could cause problems with any finite impulse response (FIR) filters used in the system. The reason for this is that the FIR filter implementation in GNU Radio uses the circular buffer for its state register. This implementation provides fast performance, however the side effect is that the state buffer is reset to zero every time the system is run, invalidating any results out of it until it refills. For a filter with many taps, this could result in an unacceptable delay from when the flow graph is run to when valid data starts appearing at the output of a filter.

Additionally, the manner in which the HARQ system has been implemented is inelegant. It relies too much on work-arounds to force the system to work in the manner intended. Ideally it would be preferable to build some higher functionality into the GNU Radio platform itself, by changing the source (as the system is open-source this is a possibility). However the amount of work required in such an undertaking puts that possibility out of the scope of this project.

11.6 Further work

There is a wide scope for further work possible with regards to the theory behind both the semi-blind LDPC turbo equalisation technique and the integration of the HARQ techniques with a practical HF system. The complexity of the turbo equalisation algorithm meant writing and debugging the code took a great deal longer than expected reducing the available time for a thorough evaluation of the techniques. These constraints also made it impossible to fully integrate the HARQ techniques with the existing system.

Additionally, a number of possible routes have been identified to build on the work presented in this chapter. They include investigating ways to improve the performance of the techniques presented and possible ways of integrating the techniques better and perhaps implementing them on different hardware to the GNU Radio platform presented here. These will be discussed in the following sections.

11.6.1 Further work on the turbo equalisation techniques

The semi-blind LDPC turbo equaliser has been implemented in the project to use BPSK only. However the theory from [87] holds for any arbitrary constellation from which it is possible to derive bit probabilities from the received signal. A useful upgrade to the system as it stands would be to adapt the existing code to use QPSK, 8PSK or, possibly, QAM waveforms. This would place the equaliser technique as a viable alternative to contemporary systems such as decision feedback equalisation which are compatible with these higher order modulations. For the purposes of finding an accurate channel estimate, Gray coding for turbo equalisers is undesirable and in fact it is better that similar valued symbols are placed as far away from each other in the constellation as possible. To this end investigations have been performed investigating new mapping techniques for turbo equalisation which may be employed with this technique in the future [45].

The packet detection procedure, which uses a short packet preamble, also generates the only channel estimate to the procedure. In the performance evaluation of the technique given in chapter 12 all the tests use a preamble length of 70 bits with a subsequent data payload of 408-2044 bits. The very low overhead for packet detection and channel probing is one of the strengths of the technique, but increasing the

amounts of probe data should increase the performance of the technique further. At the time of writing, the technique has an error floor for a given channel with no noise. This is discussed in section 12.3.2 in the evaluation. The reason for the error floor is that large changes in the phase of the received signal over the course of the data payload which the equaliser finds difficulty in adapting to, as the channel conditions become worse the error floor increases. By adding further probe data, a better initial channel estimate could be supplied to the equaliser (by, say, interpolating between probes) and this would reduce the error floor and increase the performance of the system. Any further work could investigate better configurations of probe data and packet structure to increase the performance of the technique without substantially increasing the preamble and probe overheads. Related to this, it would be useful to test this technique using standard packet structures such as MIL-STD-188-110B [20], with results from tests such as these performance comparisons could be drawn with existing techniques.

At the time of writing the only codes that have been tried with the semi-blind turbo equalisation technique are 1/2 rate LDPC codes. Intuition suggests that by increasing the code rate (thus reducing the redundancy of the code) should decrease the performance of the technique and vice-versa. An evaluation of the effects of changing the code rate on the capacity of the technique to converge on a valid channel estimate, both with and without noise, should be performed.

The high computational complexity of the technique, combined with its high parallelism, validates the possibility of implementation of the technique of an FPGA platform. Further work could focus on this, investigating fast, parallel implementations of this technique on such hardware and identifying possible optimisations. Alternatively, some work could focus on changing the algorithm code to take advantage of multi-core PCs and single-instruction multiple-data (SIMD) instructions on modern processors. New general purpose computing on graphic processor units (GP-GPU) techniques such as NVidia's Compute Unified Device Architecture (CUDA) devices could speed up the execution of these techniques by exploiting the high parallelism of modern graphics hardware. Evaluation of the technique on a high performance computing (HPC) cluster would also be interesting.

This technique could also find application in other communications systems outside of the HF world. By reducing the overhead of any packet structure throughput of any communications channel could be increased. As LDPC techniques start seeing uptake in commercial systems, integration of techniques such as this become a

viable option for channel equalisation. For very high speed systems, however, this technique may be to too computationally complex for contemporary processors.

11.6.2 Further work on the HARQ techniques

The problem of implementing the HARQ techniques discussed above with a working HF system have not yet been satisfactorily resolved. There is work required to integrate these techniques with GNU Radio, if it is desired that the platform is to be used in further investigations. There is also work required to be done on integration of the HARQ techniques with the turbo equalisation algorithm, including finding a good way to equalise subsequently received repeat packets to a rapidly changing channel.

It may be the case that the new HARQ technique described in section 7.10 may be a poor choice of repeat request technique to use with turbo equalisation. It may be the case that simply repeating the full LDPC packet may be the simplest and most robust way of implementing ARQ on the communications link defined in this project. This, however does not detract from the fact that the HARQ techniques investigated are interesting in their own right, and the new RSE-HARQ technique could be used in other systems, where equalisation with a channel is not such a major obstacle. It may be possible to integrate the new technique with OFDM systems or other non-HF systems.

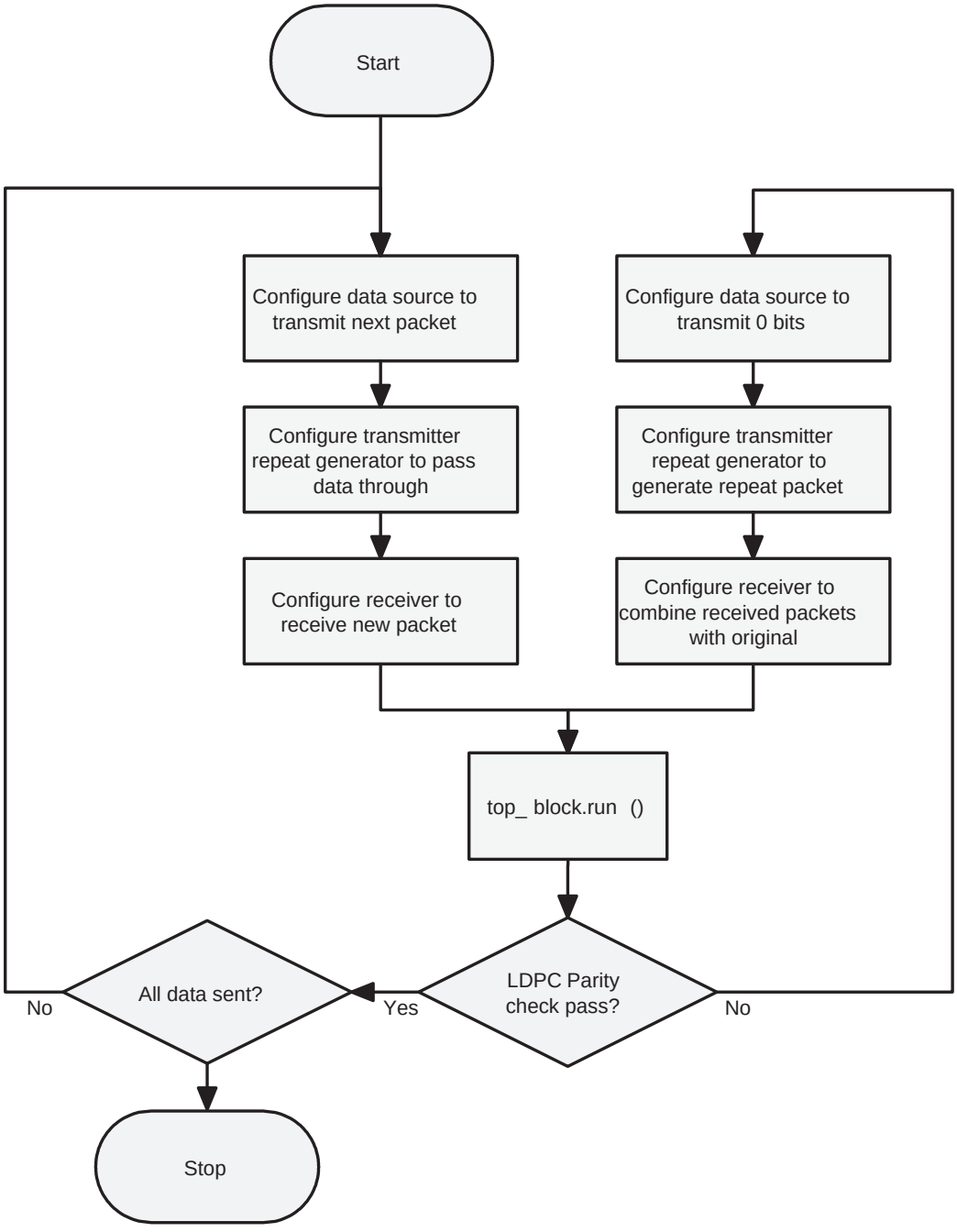


Figure 11.5: Algorithm used to implement HARQ techniques

Chapter 12

System performance evaluation

This chapter describes the evaluation of the techniques used within the proposed system. From chapter 4, the aim of the system is to efficiently (in terms of power required per successfully received bit) send data a long distance from a remote, battery operated unit to a complex, mains powered base station. The data transfer is primarily in one direction, from the remote unit to the base station. This asymmetry entails some particular performance requirements. The sections within this chapter attempt to characterise the techniques used, and provide some basic understanding of their performance and appropriate ways to integrate them into a final system.

All of the performance tests were performed on the GNU Radio platform. A complete system was simulated on one computer without using any transmitter hardware such as the universal software radio peripheral (USRP). This platform was seen to be a useful tool for the evaluation of such techniques due to the ease with which a system could be modified and the speed of the platform compared to, say, Matlab. The platform is not, however, optimised for use with testing systems and there were a few problems encountered when implementing the simulations. These are described within the text. For further details of the use of GNU Radio for these purposes refer to [113].

12.1 Error rate performance

This section details the performance of the error control systems under AWGN. The error correction systems investigated are low density parity check (LDPC) based

forward error correction (FEC) and hybrid automatic repeat request (HARQ) techniques. These techniques are described in chapter 7, and the implementation of them is described in chapters 11 and 10. The LDPC FEC system is additionally used to aid the channel equaliser generate accurate channel estimates using a technique known as LDPC turbo equalisation which is described in chapter 8, with details of the implementation in chapter 11. The extra degrees of freedom inherent in the use of this technique may have an impact on the bit error rate (BER) performance of the LDPC code under simple AWGN channels, or other situations where a perfect channel knowledge is assumed.

12.2 FEC performance under AWGN

The simplest channel used to analyse bit error rate performance is the additive white Gaussian noise (AWGN) channel, which is a simple memoryless channel where normally distributed noise samples are added to each sample passing through the channel [3]. This channel is widely found in the literature, and as such is suitable to use for comparative evaluations of communication systems. A metric commonly used to define the signal to noise ratio (SNR) of a digital signal is E_b/N_0, where E_b is the energy of one bit in the system and N_0 is the noise spectral density. This measure allows comparisons of error correction systems independently of sampling frequency, signal bandwidth and the modulation technique used. E_b/N_0 is related to the linear SNR by

$$\text{SNR} = \frac{E_b}{N_0} \times \frac{f_b}{B}, \tag{12.1}$$

where f_b is the data rate of the system and B is the signal bandwidth.

The code designed for use with this project is a 1/2 rate QC-LDPC code with codeword length of $n = 2044$ (see section 10.4.6.1). Additionally many tests have been performed using a shorter $n = 408$ 1/2 rate random LDPC code which has been designed and made available by David Mackay on his website [128]. Two other 1/2 rate random LDPC codes with $n = 204$ and $n = 816$ have also been tested to provide a comparison between different length LDPC codes.

To perform the tests a simulation was set up using GNU Radio. A random bit source which sends bits generated by a pseudorandom noise (PN) generator was used as the data source. This source data is then fed into an LDPC encoder and modulated

into QPSK symbols. AWGN is simulated by adding random, Gaussian distributed noise with variance ν, where

$$\nu = -E_b 2N_0. \tag{12.2}$$

The output of the channel was then fed into an LDPC decoder, which outputs just the received message bits to a 'bit sink' processing block. The decoder was set to use a maximum of 50 iterations to attempt decode the packet, after the 50 iterations completed if the packet had not yet been successfully decoded the current 'best guess' of the codeword was used and its message bits forwarded to the bit sink. The bit sink's function was to check the number of errors in its received vector by comparing them with its own, synchronised PN generator. Figure 12.1 gives a block diagram of the setup used to perform the tests. The data out of the LDPC encoder is modulated using binary phase shift keying (BPSK) before noise is added, then demodulated using a soft-output BPSK demodulator [10]. This demodulator simply outputs the log-likelihood ratio of the likelihoods of each received bit to the decoder, as BPSK is used, the symbol likelihood is equivalent to the bit likelihood and is given for a received symbol, y, as

$$L(y = a) = \frac{e^{\frac{|y-a|^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}} \tag{12.3}$$

where σ^2 is the noise variance and $a \in \mathcal{A}$ is a BPSK symbol drawn from the set $\mathcal{A} = \{1, -1\}$. The output of such a decoder in the case of BPSK is therefore

$$\text{LLR} = log\left(\frac{L(y = 1)}{L(y = -1)}\right). \tag{12.4}$$

It is evident that an estimate of the noise variance, σ^2 must be given to the soft demodulator block, this is trivial when simulating as the variance used to generate the noise can be used, in a practical system some means of finding this from the received signal would have to be considered.

For each test both the raw bit error rate and the frame error rate were found. The bit error rate being the probability that any individual bit is in error, and the frame error rate being the probability that any received packet is found to contain at least one error.

The results show a clear increase in code performance as the codeword length increases, the $n = 2044$ quasi-cyclic LDPC (QC-LDPC) code exhibits the best results, requiring only 3 dB SNR at the receiver to decode with a very low error rate. Inter-

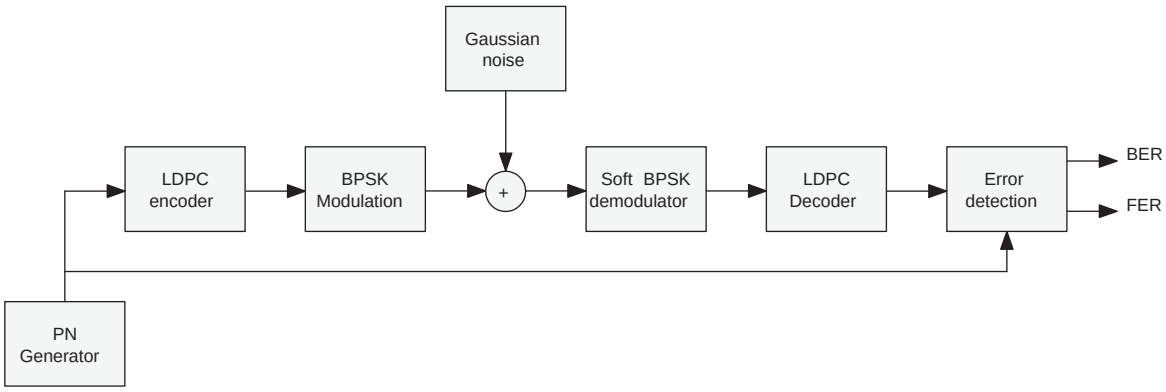


Figure 12.1: Simulation set-up for LDPC AWGN tests

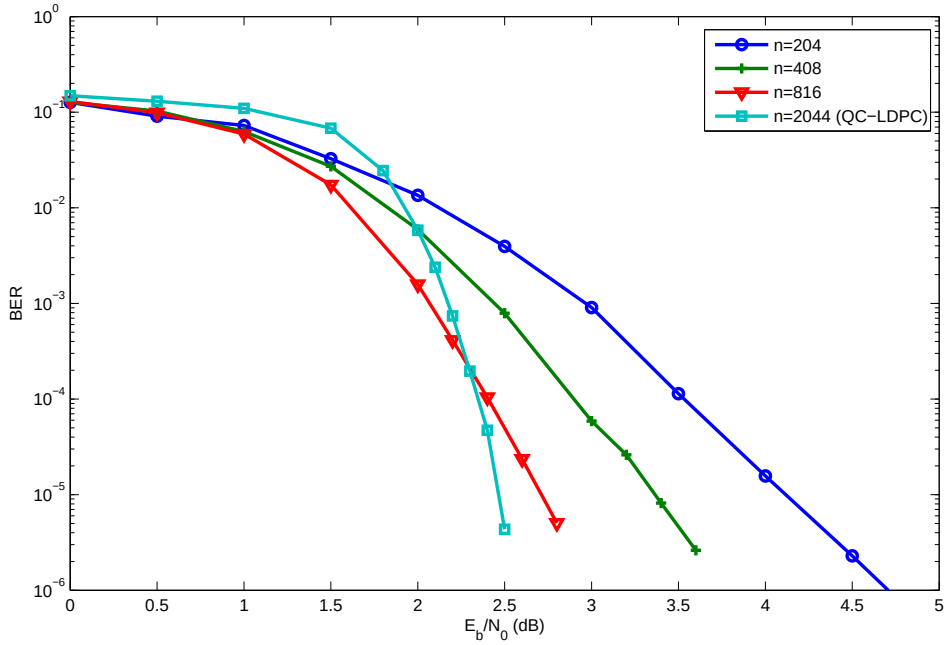


Figure 12.2: Bit error rate (BER) of 1/2 rate LDPC codes

estingly the $n = 2044$ bit QC-LDPC curves cross the other curves, exhibiting very poor error performance at $E_b/N_0 < 1.5$ dB but quickly outperforming the other codes. The higher performance is expected as the code length is longer, however the poorer observed performance at very low SNR exposes a property of the QC-LDPC code compared to the other codes (which were all random Mackay types).

12.3 Semi-blind LDPC turbo equaliser performance

The semi-blind LDPC equaliser is described in chapter 8. It is a modern, complex equalisation technique which is capable of adapting itself to an arbitrary channel

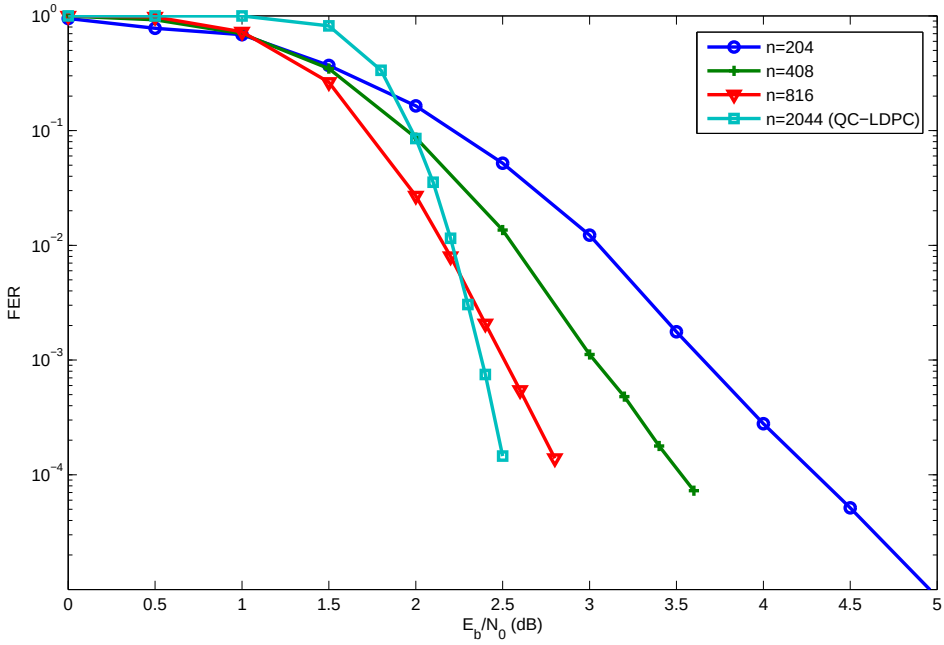


Figure 12.3: Fframe error rate (FER) of 1/2 rate LDPC codes

given an initial channel and noise variance estimate. It is also very computationally complex to run, requiring considerable processing power and memory to run effectively. The algorithm is parallelisable, however, and so can be implemented on multi-core computers or clusters, or potentially graphics cards (using CUDA, for example). For this reason some of the results graphs are not smooth (there is some statistical variation in the data points) and it was impossible to perform tests on some of the poorer channels. The following results, however, provide a useful insight into the baseline performance of the technique and show it to be a compelling addition to any modern or future HF radio communications system.

In all test cases BPSK was used to modulate the data. The reason for this is that it simplifies the operation of the equaliser and requires less processing resources than using a higher order modulation technique such as QPSK or QAM. The technique is, however, capable of working with any given modulation constellation with some minor changes to the existing code [87], and this should certainly be considered for use with any subsequent work.

12.3.1 Convergence of the channel estimate

The turbo equalisation technique adapts an internal channel estimate to the received data. This adaptation occurs over the course of multiple iterations of the algorithm,

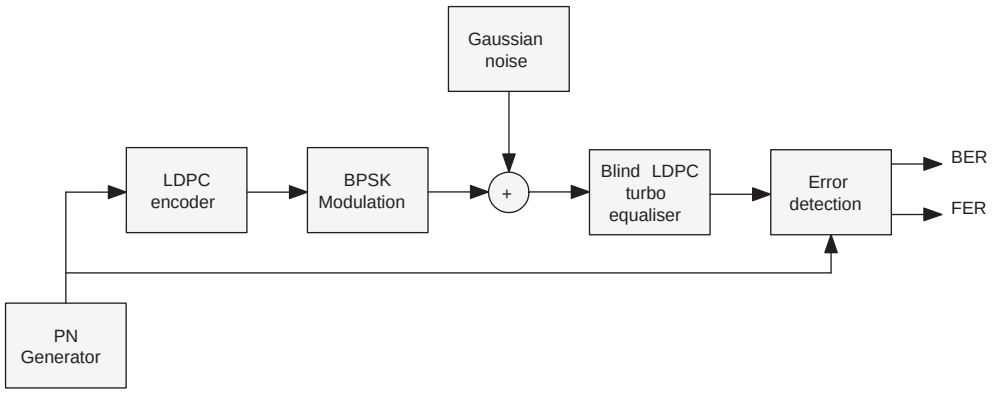


Figure 12.4: Simulation set-up for semi-blind LDPC turbo equaliser AWGN tests

and the estimated channel taps can be observed closing in on the real taps used to define the simulated channel. It is interesting to see these iterations, as well as useful to measure the speed at which they converge. To these ends, a test was performed to demonstrate the convergence of these taps to the real channel taps.

A simulation was set up in GNU Radio where an LDPC encoder was used to encode a vector of random bits into a codeword. This was then fed through a noiseless channel with channel response $\mathbf{h} = \{1.2, 1.2, 0.45, 0.6\}$. The output of the channel was fed into an semi-blind LDPC equaliser. An initial channel estimate was provided to the equaliser of $\mathbf{h}_e = \{1, 0, 0, 0\}$. The simulation was then left to run until the LDPC codeword was successfully decoded (at which point the channel estimator is guaranteed to give a perfect estimate of the channel, due to it being noiseless). This process was carried out 100 times and the mean of the channel estimates at each location were taken.

Figure 12.5 shows the mean value of the channel estimate at each iteration in the turbo equalisation process. From this graph it is apparent that within 10 iterations the channel estimate will generally have converged upon the correct values. This also demonstrates the power of the technique to converge correctly upon rapidly changing channel parameters.

12.3.2 Tracking the ITU recommended channels without noise

It is interesting to observe the operation of the semi-blind LDPC turbo equaliser as it tracks the changing channel taps of the simulated HF channel. Simulations were performed, similar to the noise test described in the next section (see Figure 12.11),

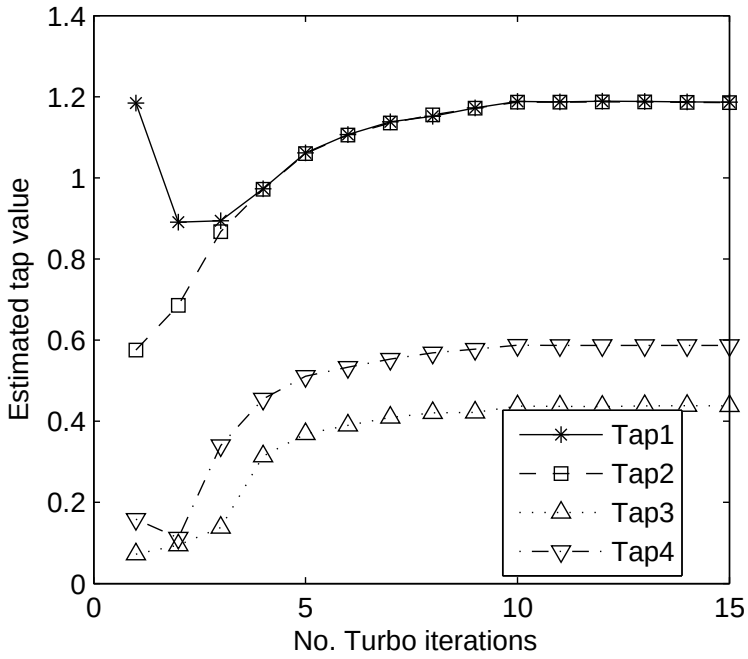


Figure 12.5: Convergence of channel estimate in a noiseless channel

but without AWGN. For these simulations the ITU-R recommended channels were used, which are described in section 2.6.

The test was set up with LDPC encoded data being sent through a channel at 4 kbps, using the 2044-bit 1/2 rate LDPC code defined in section 10.4.6.1. Within each turbo iteration the LDPC decoder ran for a maximum of 50 iterations and up to 30 turbo iterations were run for each packet. If after 30 turbo iterations the packet was still not decoded, the best estimate for that packet was recorded and the next packet was sent through the channel. A 63-bit M-sequence was used as a preamble to each packet. This wasn't used to synchronise the packets, as fades could occasionally cause the synchronisation to fail. The probe data was used to provide information to the receiver that allowed it to find an initial channel estimate for each packet. Figure 12.6 shows the tracking of the CCIR good channel. At the simulated data rate of 4 kbps (after FEC coding) the estimated channel length is appropriately represented by a FSR of length 3. The first and last registers of the FSR were recorded as the centre register was assumed to be low (as it doesn't correspond to any simulated channel tap). For each packet, there were 5 channel estimates generated that were spread evenly in time along the length of the packet. Figure 12.7 shows the mean squared error of the turbo equaliser tracking the channel. The errors are small on this, the comb-like distribution of errors is due to the lower resolution of the estimates compared to the sampling rate of the system.

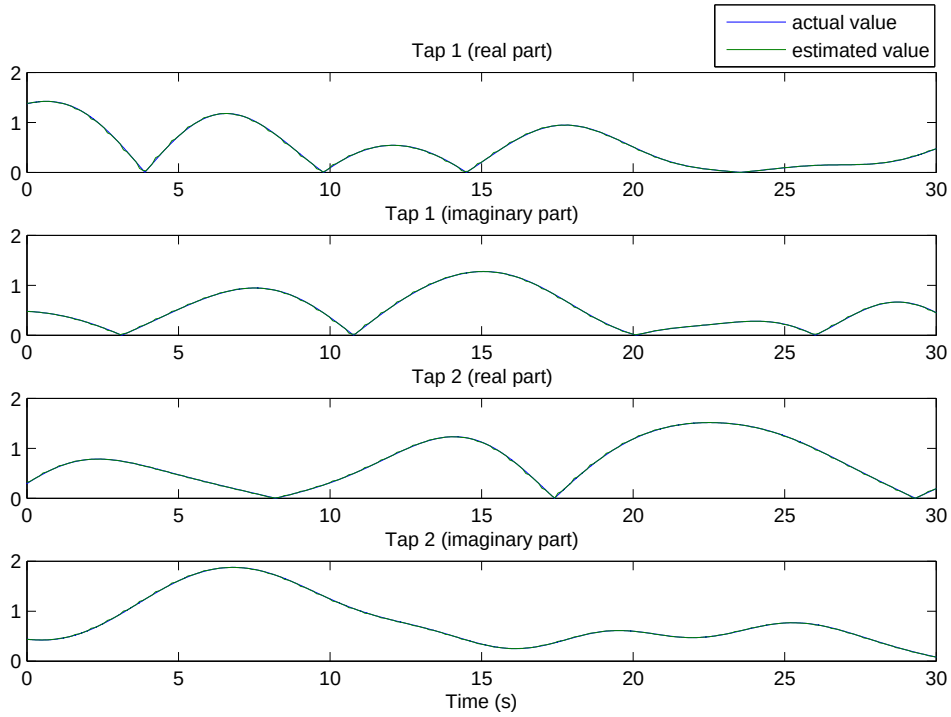


Figure 12.6: Semi-blind turbo equaliser tracking the ITU-R good channel (without noise)

Figure 12.8 shows the semi-blind turbo equaliser tracking the ITU-R moderate channel, and Figure 12.9 shows the mean squared error of the path estimates. This simulation was run identically to the simulation above, however now the FSR length is set to 5 as the multipath delay is doubled with respect to the ITU-R good channel, at a time of 1 ms.

The ITU-R good channel test is seen to be very successful, indeed during the course of this simulation no errors were observed. The estimated channel is virtually indistinguishable from the actual channel taps. The ITU-R moderate channel test is still successful, but in this test errors can be observed in the results. These tend to be close to deep fades, where the channel taps are changing fastest. As Doppler spreads increase beyond 0.5 Hz, the performance of the equaliser is seen to rapidly decrease for the 2044-bit QC-LDPC code for the parameters given in this simulation. By increasing the data throughput of the system (and therefore decreasing the amount that the channel changes over the course of a packet), the performance can be increased somewhat, but in practice this will cause the SNR of the transmission to decrease (by increasing the bandwidth of the transmitted signal), and thus increase the noise related error rate. Given the hardware constraints of the evaluation computer it was impossible to model the ITU-R poor channel, as the memory

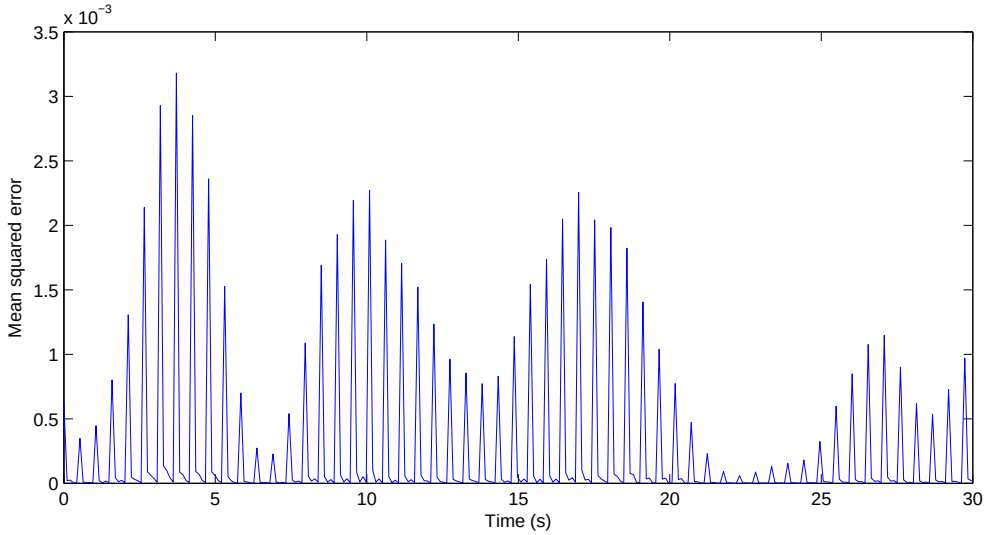


Figure 12.7: Semi-blind turbo equaliser tracking the ITU-R good channel (mean squared error)

requirements for the matrix of pairwise probabilities was too large to fit into the memory. To implement a decoder on this channel a computer running a 64-bit operating system with a large amount of RAM is required.

As there are errors in the system even without noise, it was decided that it would be worthwhile to test what the noiseless error rate of the technique was for a range of different spreads. The parameters of the test were the same as the ITU-R good channel test above. That is, the bit rate was kept the same (4 kbps after FEC encoding) and the channel was left as a two path channel with a multipath delay of 0.5 ms. The Doppler spread for both paths was varied through a range between 0.1 Hz and 1 Hz (both paths used the same value of spread). Figure 12.10 shows the results of these tests on the 2044-bit QC-LDPC code and the shorter 408-bit random LDPC code for comparison. There is a large performance increase with the shorter code over the longer code. This is due to the fact that the channel varies less relative to the entire LDPC codeword as the codeword decreases in size. This effectively represents an increase in the frequency that probe data is sent down the channel with the LDPC encoded data payload. This suggests that sending more probe data will have a strong effect on the performance of the equaliser.

The above graph shows that there is a minimum bit error rate for a given channel with the set up used in this simulation. Improvements in these bit error rates should be possible by using a longer probe sequence or otherwise improving the channel estimate. The set up above uses 63 bits of probe data as a preamble to a 2044 bit

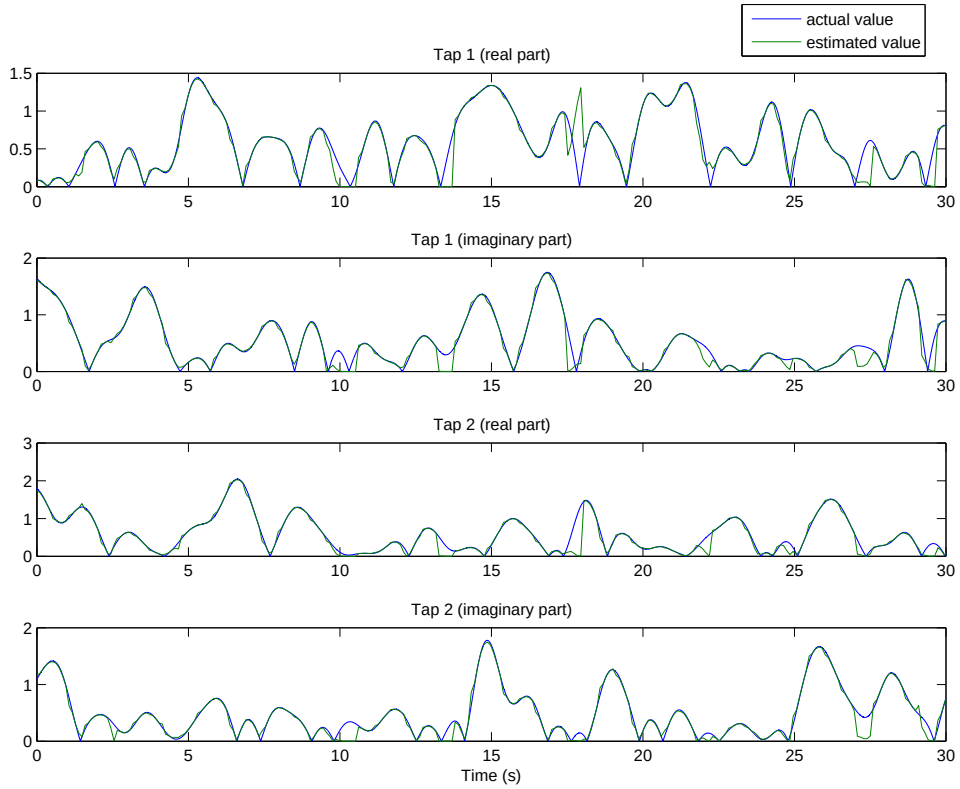


Figure 12.8: Semi-blind turbo equaliser tracking the ITU-R moderate channel (without noise)

data payload, this represents a very low overhead of approximately 3% for probe data. As it is conFigured at the moment, the channel correlator is only supplying a single estimate to the channel estimator for the whole ensuing packet. This should be improved such that the estimates for the probe data both before and after the payload are taken and the initial estimates supplied to the channel equaliser interpolate between them. Further increasing the probe data should allow better prior knowledge of the channel to be supplied to the equaliser and thus increase performance.

12.3.3 Error rates with ITU-R channels and AWGN

The noise performance of the semi-blind turbo equaliser was assessed using the ITU-R recommended channels. The simulations were conFigured according to the block diagram in Figure 12.11. For every packet a PN sequence is generated as the data payload. This is then LDPC encoded, BPSK modulated and a 63-bit packet header is added. The packet is then sent through a simulated ITU-R channel. The

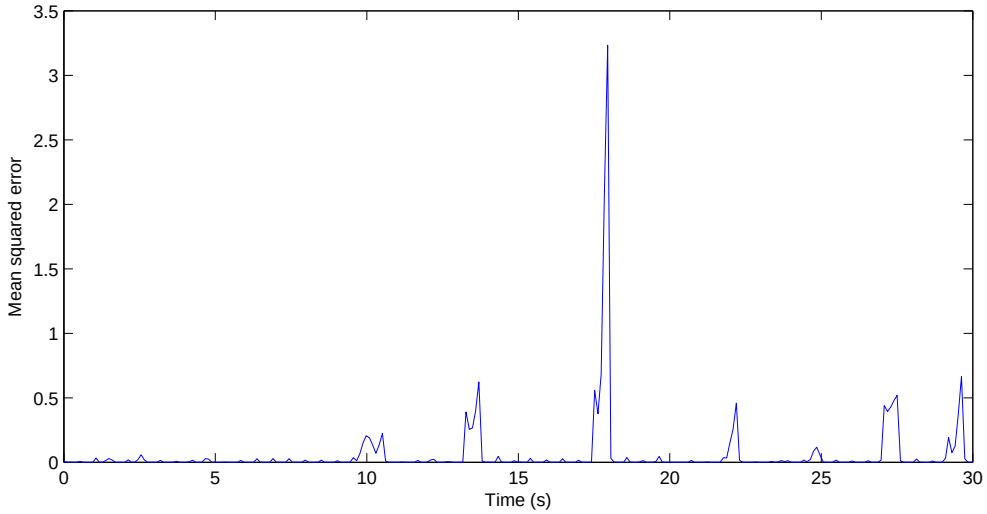


Figure 12.9: Semi-blind turbo equaliser tracking the ITU-R good channel (mean squared error)

output of the channel is fed into a correlator which finds the cross correlation of the received header and the original header pattern to find an initial channel estimate. The header is stripped from the received packet and fed into the LDPC turbo equaliser together with the channel estimate. The turbo equaliser is conFigured to use the same number of iterations as the tests above, namely 50 iterations of the BP algorithm for decoding the LDPC packet and 30 turbo iterations to equalise the packet. If after the 30 turbo iterations the LDPC packet has not been successfully decoded the best estimate is forwarded to an error detector which detects bit and packet error rates.

The reason that the same PN sequence was used as the data payload was to simplify the implementation of the test script. No sequence numbers were added to the packet and the way in which GNU Radio is implemented makes the comparison of the received packets with the correct data rather difficult. This may introduce bias into the LDPC decoder performance, but it was decided that this bias was unlikely to be significant, especially given the fact that only the equaliser was being tested, and not the performance of the embedded LDPC decoder (which is assessed above).

Figures 12.12 and 12.13 show the results of the bit error rate tests for the ITU-R good and moderate channels respectively. The computational capacity and memory of the test machine was too low to perform the equalisation tests on the ITU-R poor channel. The tests were performed with the 2044-bit QC-LDPC code and the 408-bit random LDPC code. For the 2044-bit LDPC code the bit error rate con-

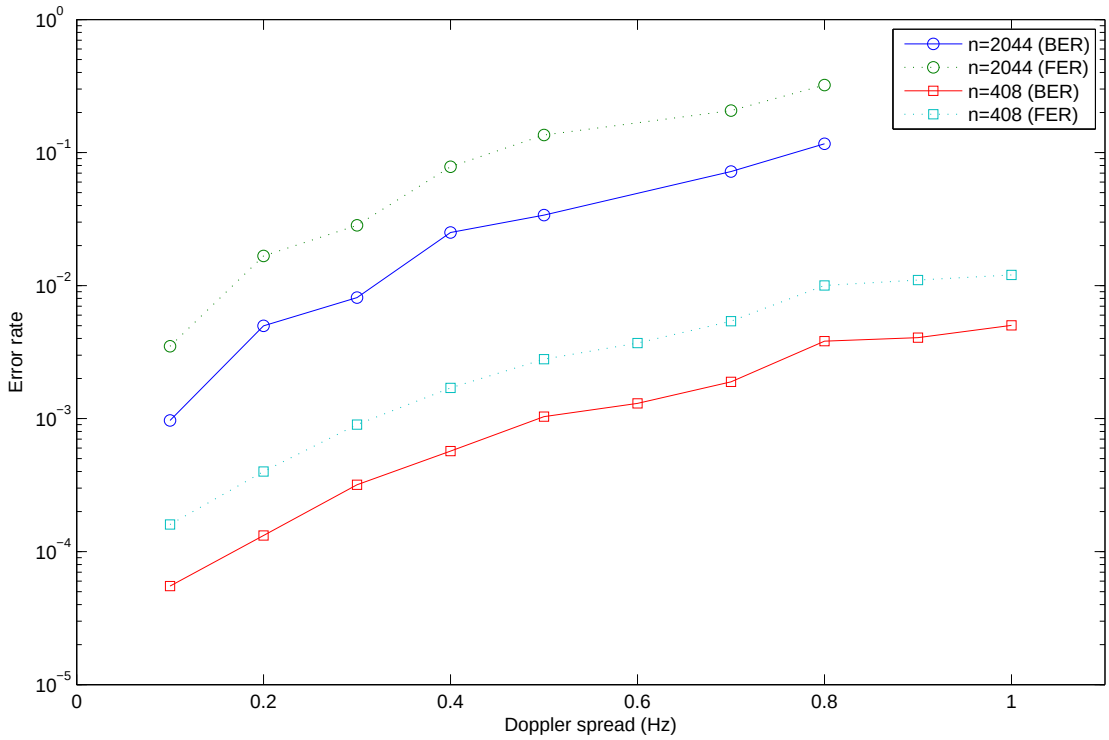


Figure 12.10: Error rate compared to Doppler spread of a two-path channel with no noise

verges on approximately 0.01 and just over 0.02 for the good and moderate channels respectively and the packet error rate converges on approximately 0.03 and 0.08 respectively. These Figures seem to suggest that there will be approximately a 3% and 9% overhead (not including header data) if an ARQ technique was used to repeat the packets. The 408-bit codeword had lower error rates all round corresponding to an approximately 1% and 2% overhead for the good and moderate channels respectively. These results will be discussed below in section 12.6.

12.3.4 Improving the performance of the equaliser

It appears to be a lot of scope for optimising the equaliser to increase performance beyond the levels given above. As previously stated, by increasing the accuracy of the initial channel estimates supplied to the equaliser before it attempts to equalise and decode an incoming packet, it is likely that an increased likelihood on converging to the correct channel should result. This should lower the error floors for the above tests and perhaps allow higher Doppler spreads to be used (so that, memory allowing, the ITU-R poor channel could be properly evaluated). The initial channel estimates could be increased in accuracy by increasing the amount of training data

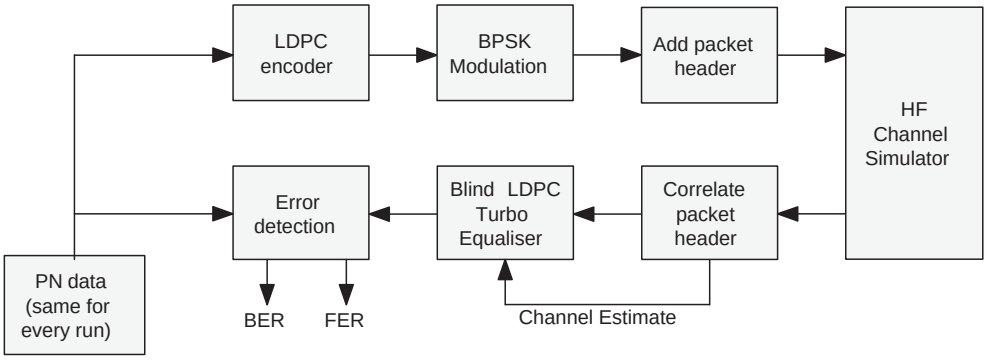


Figure 12.11: Set up for evaluating the semi-blind turbo equaliser noise performance evaluation on the ITU-R channels

used for each packet from the very low level used presently and interpolated between to form new estimates. It is also possible that channel estimates could be generated for a given packet by linearly interpolating between the header of that packet and the next, a technique which is not currently being used and should allow better performance with no additional overhead. It would also be an interesting test to build this equaliser into the MIL-STD-188-110B waveform [20] (see section 3.1), which provides a standard level of overhead, much higher than that used for the tests above, so that the new technique can be properly compared with the many other techniques that have been investigated in the literature for use with this waveform.

Another standard means of compensating for fading in communications channels is by using an interleaver in the transmitter, and a corresponding deinterleaver in the receiver. It may be the case that inserting an interleaver into the system to assess whether any improvement to the performance results. However as the structure of LDPC codes is similar to an equaliser, and spreads the effects of fades across the length of the codeword it is unlikely that any great improvements in performance will be observed.

As the LDPC code length increases, from Figures 12.12 and 12.13, it appears that the semi-blind equaliser becomes less capable of equalising fading channels. This is due to a higher likelihood of deep fades over the length of the codeword which are difficult for the equaliser to compensate for. A potential remedy for this would be to add extra probe sequences regularly throughout the length of the codeword. This would have the effect of splitting the codeword into smaller sections. Between each section a channel estimate could be generated and fed into the algorithm. By adding this extra probe information the higher performance in fading channels of

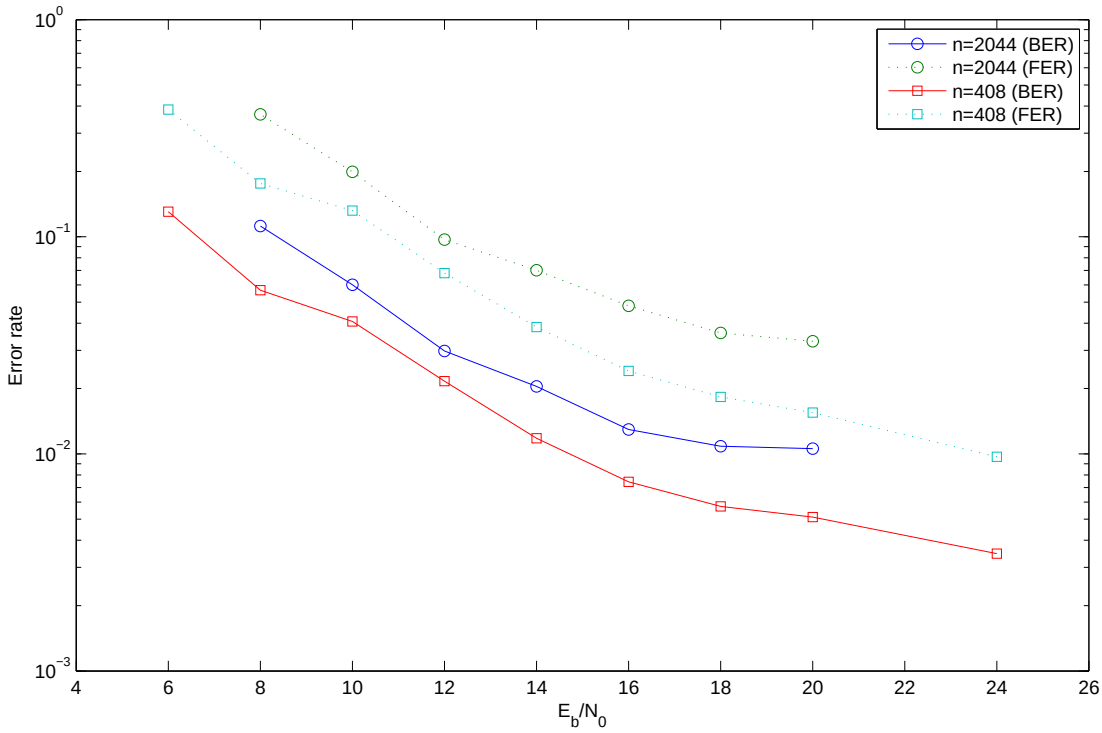


Figure 12.12: Error performance of the LDPC turbo equaliser in the ITU-R good channel

short codeword lengths could be combined with the better noise performance of longer LDPC codewords, at the expense of a small increase in overhead.

12.4 Issue identified with QC-LDPC codes and semi-blind turbo equalisation

During the performance evaluation, an issue was discovered relating to the use of QC-LDPC codes in semi-blind LDPC turbo equalisers. The issue appeared when a received vector of symbols representative of a BPSK modulated QC-LDPC codeword, having been corrupted by a channel with memory was fed into a semi-blind turbo equaliser and decoded. If the provided channel estimate was not sufficiently good, the LDPC decoder embedded in the semi-blind equaliser would occasionally output a code which appeared feasible and correct (i.e. it has the expected number of 1-bits in it and there were no major errors seen in decoding), but would be detected as having a large number of errors in it by the error detector (a BER of 50%).

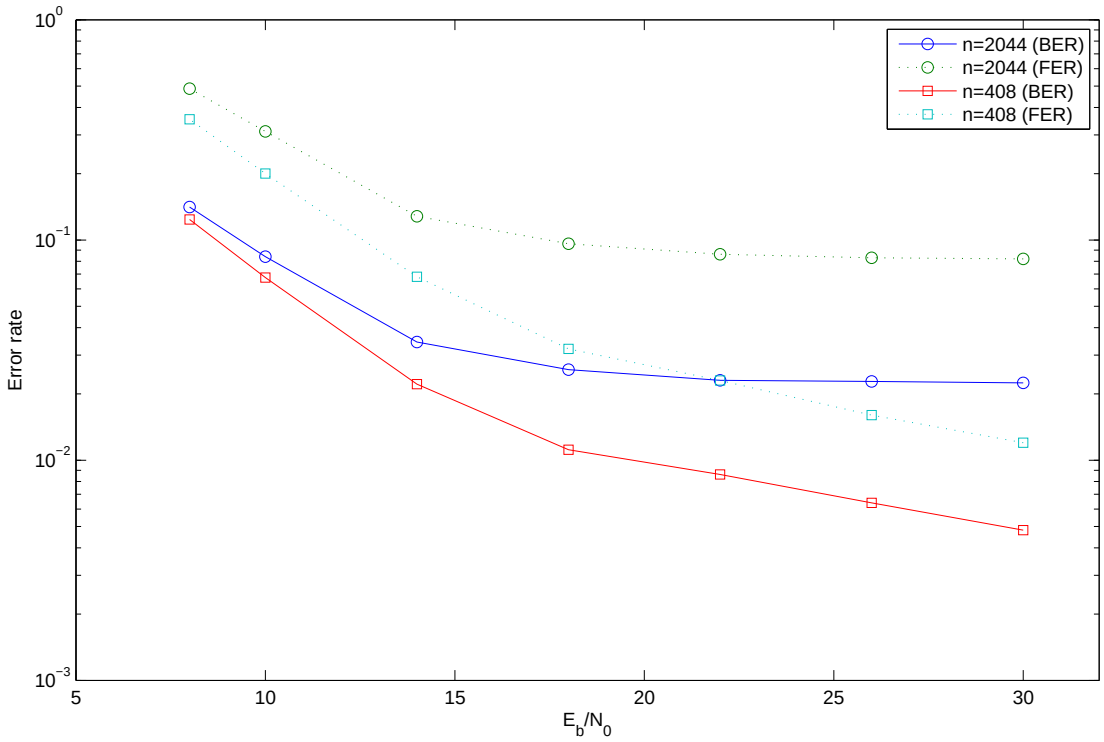


Figure 12.13: Error performance of the LDPC turbo equaliser in the ITU-R moderate channel

The reason for this was established as being that due to the quasi-cyclic nature of a QC-LDPC code. If any given codeword is shifted a number of bits left or right, a sequence is generated which is *similar* to another, valid codeword. In this case *similar* refers to it having a low Hamming distance to another valid code. Indeed for a QC-LDPC code constructed of P cyclic sub-matrices, with a code length of $n = rP$, where are is a positive integer, if any valid codeword is split into r P-length sections, each of these sections are rotated left or right by any number of bits and the new sequences are concatenated again, the result is another valid codeword. This property is due to the construction of the codes (see section 7.5.2 for further details on the structure of QC-LDPC codes).

The property in the above paragraph, in combination with the manner of operation of the turbo equaliser, which attempts to converge to the closest solution of the problem, which is finding a valid combination of channel estimate and valid LDPC code, can mean that the result of the operation converges upon the incorrect channel estimate, but which still allows the QC-LDPC code to fulfil its parity check equation. The symptom of this problem is that the LDPC code looks valid, but the estimated channel taps may be in the wrong position. So for example, if a QC-LDPC codeword is received through a channel with taps $\{$1.2+0.6j, 0.3+0.4j$\}$, the codeword might be

decoded, fulfil its parity check and give a channel estimate of $\{0+0j, 1.2+0.6j\}$, the LDPC decoder has converged to the incorrect, quasi-cyclically rotated version of the original word and the second, lower magnitude path of the channel has been treated simply as noise in the decoder.

This problem is not a major issue when a sufficiently good channel estimate is provided to the LDPC equaliser, but under low SNR conditions or rapidly changing channel conditions it could become problematic. To solve this issue it is proposed that either a random LDPC code be used (which is not an ideal solution if the encoder is to be a simple, low power device, see section 10.4.6), or a scrambler be used to scramble and unscramble the LDPC code (which may be feasible but would require some changes to the LDPC turbo equaliser algorithm to be implemented).

12.5 HARQ performance under AWGN

The HARQ techniques described in section 7.7 were also tested under additive white Gaussian noise (AWGN), to determine their throughput performance. Tests were performed using the 1/2 rate LDPC code evaluated above in section 12.2 with the codeword length of 408, of which the message length is 204. To perform the tests a 6528-bit random message was generated and copied as a binary file into the computer's file system. This was used as a bit source for the throughput test, which was performed in GNU Radio. The length of the random message was chosen as an integer multiple of the message length ($204 \times 32 = 6528$). Each test was performed for a range of values of SNR, and run a total of 50 times for any given SNR. The test results at each SNR are the mean values across all 50 of the tests performed at that level.

Each test involved setting up a GNU Radio flow graph which included the transmitter, AWGN channel and receiver. The transmitter code read data in from a file, encoded it using LDPC and sent it through the channel to the receiver. Upon receiving a packet the receiver attempted to decode the LDPC packet, if decoding was unsuccessful a request was sent back to the transmitter and the transmitter would generate a repeat packet for the next transmission. For all successfully decoded packets at the receiver the result was copied to an output file, it is important to note here that a 'successful' decode was assumed to be one in which the result of the LDPC decoding operation passed its parity check. A record of how many bits in total

had been sent through the channel was kept and used with the known packet length to generate throughput ratios. For more details on the implementation of the HARQ throughput simulations refer to section 11.5.2.

For each simulation the LDPC decoders were set to declare decoding a failure if it had not been achieved within 50 iterations. When sending a packet for the first time the entire LDPC codeword, that is 408 bits, was sent. Upon each repeat request a further 40 bits were sent through the channel, therefore every subsequent repeat that was sent gave approximately an extra 10% incremental redundancy to the transmission. The RSE-HARQ used combinations of two bits to generate each repeat bit, this was found to give good overall results, but the effects of varying the degree of the RSE-HARQ packets will be discussed further below.

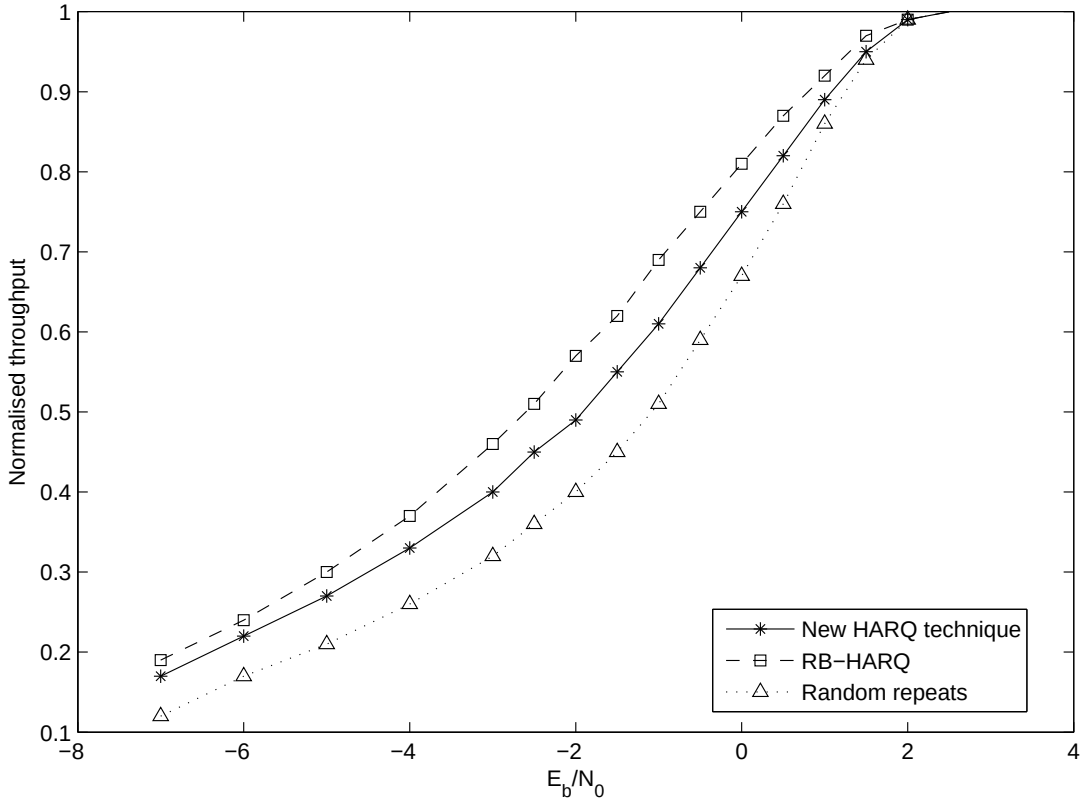


Figure 12.14: Throughput of the three HARQ techniques under AWGN

Figure 12.14 shows a graph comparing the three techniques, the graph shows that RB-HARQ performs the best out of the three, with the random repeats technique performing the worst. The new technique, RSE-HARQ, has performance between these two techniques, outperforming the random repeat technique but not reaching the performance of RB-HARQ. As RSE-HARQ does not require a wide-bandwidth,

error free return path, however, it may be deemed a more appropriate choice of technique for applications such as the one discussed in this project. With non-incremental chase-combining, where the entire packet is resent when an error is detected in the original, the graph would show discrete throughput levels with values of $1/N$, with N a positive integer. It is clear that the incremental techniques will outperform chase combining at higher E_b/N_0 levels (over about -2 dB).

12.5.1 Varying the RSE-HARQ set size

As mentioned in section 7.10.2 the set size chosen for RSE-HARQ has an effect on the throughput of the technique. A simulation was performed to show the effect of varying the set size. The set size refers to the number of packets (or bits) combined to form each repeat packet (or bit). The simulations above all used a set size of two. The simulation was set up the same as the RSE-HARQ simulation above, and finds the throughput of the technique by finding how many bits are required to receive the correct data.

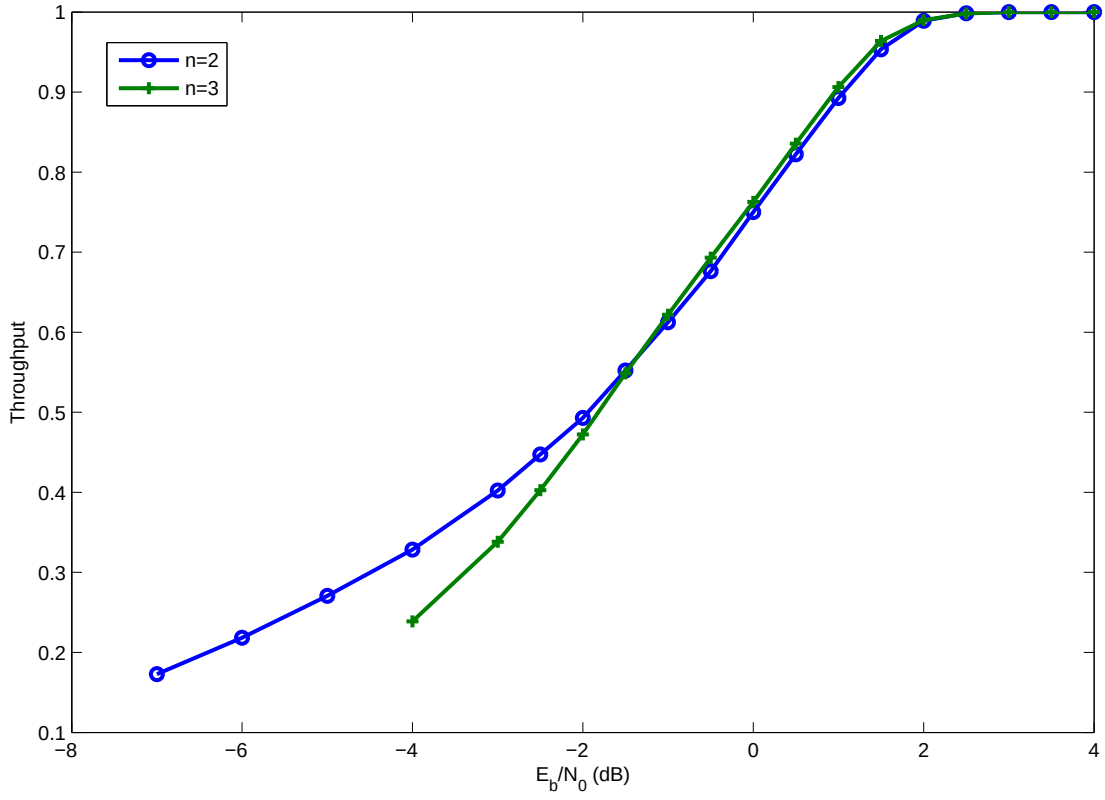


Figure 12.15: Effects on throughput of varying the RSE-HARQ set size

Figure 12.15 shows the results of changing the RSE-HARQ set size on the through-

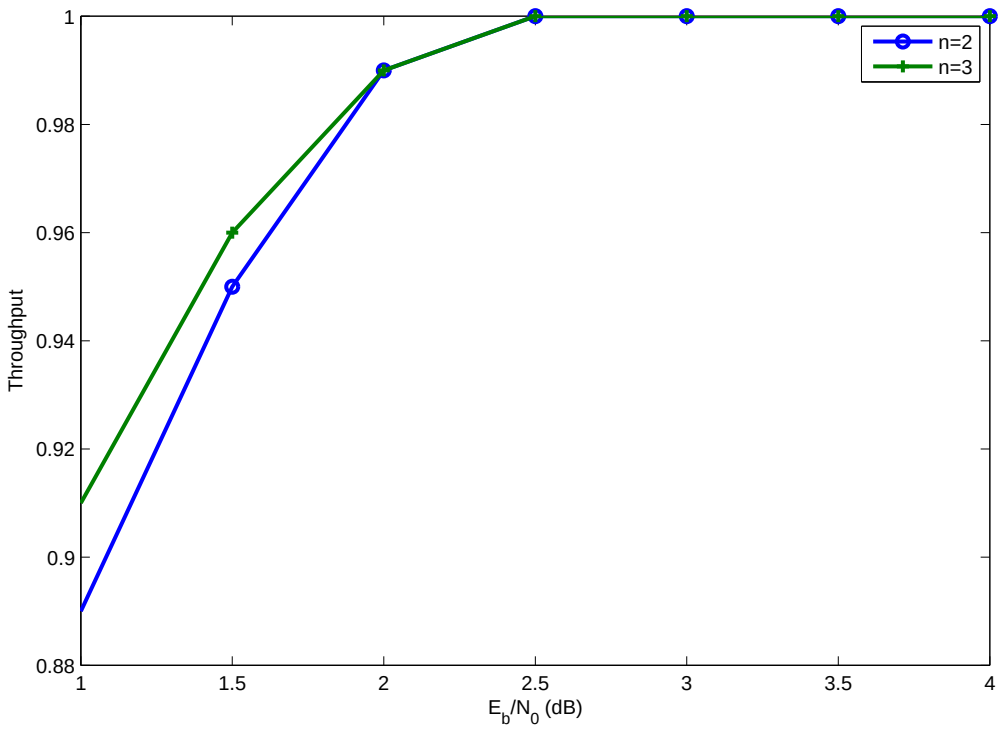


Figure 12.16: Effects on throughput of varying the RSE-HARQ set size (zoomed)

put of the system. It is apparent that at lower SNRs, it is preferable to have a smaller set size as the throughput is increased. However at higher SNRs, where the error rate is low and the throughput is only just starting to decrease, using a higher set size appears to give better performance results. This is exhibited in the zoomed version of the graph in Figure 12.16, where it is apparent that the better performance of the RSE-HARQ technique is observed with a set size of $n = 3$. The reason for this behaviour is most likely that, at low SNRs, increasing the set size gives the technique a higher chance of covering a bit decoded with a low posterior likelihood compared to a smaller set size. At higher SNRs, however, the extra degrees of freedom arising from using the larger set sizes counts against the performance of the technique, resulting in a faster rolloff of throughput performance.

It would be worth examining this property of the technique in further work to find a good, adaptive method of changing the RSE-HARQ set size through varying channel noise conditions. This might take the form of resending the first few packets with a high degree set size and decreasing this as the number of repeats for a single packet increases.

12.6 Throughput performance

From the results given above, by using simple repeats of incorrectly received packets, the overhead, not including the overhead of the FEC coding used, for the CCIR good and CCIR moderate channels are approximately 6% and 13% respectively. This compares favourably to the approximately 25% overhead from using the industry standard MIL-STD-188-110B waveform with [20] with short burst transmissions (see section 3.1).

The error floor of the technique is currently rather higher than might be expected in a communications system, but by implementing the improvements suggested in section 12.3.4 it will be possible to reduce these, in which case the curves in error rate curves in Figures 12.12 and 12.13 will show improved performance. Assuming that the error rate decreases at the same rate that it appears to between 10 and 15 dB points in Figure12.12 and 12.13 this technique shows very high performance for low packet overhead compared to current communication systems.

From Figures 12.12 and 12.13 it appears that the shorter, 408 bit LDPC code is performing better than the longer 2044 bit QC-LDPC code. This is misleading, however, as there is clearly more probe data sent with the 408 bit code than there is with the 2044 bit code. The probe data overhead (which is one 63-bit preamble sent before each codeword) for the 2044 bit code is approximately 3% and the overhead for the 408 bit code is approximately 13.5%. These results are actually showing the potential increase in bit error rate with an associated increase in probe data sent with the data.

12.7 Further work

There are further tests that can be performed on the individual algorithms, and entire system, which have not yet been performed due to limited time constraints. There are many different parameters to change in the system which may have an effect on its operation and throughput performance. The system itself has not yet had any proper throughput evaluations. The main reason for this being the unsolved integration issues with GNU Radio as described in section 11.5.3.1.

It would certainly be worth assessing how the performance of the equaliser changes

with different LDPC codes. Changes to the rate, codeword length and structure of the LDPC code should all be examined with respect to how they change the convergence of the equaliser to the channel. Additionally it may be worth finding optimal values for the maximum number of iterations parameters used in the LDPC decoder and semi-blind turbo equaliser. Currently, using 30 turbo iterations appears to be adding unnecessary processing overhead to the equalisation process in the case where the decoder cannot find the correct LDPC code.

It may be worthwhile attempting to find a sound mathematical basis for the performance increase seen for the novel RSE-HARQ technique. This should take into consideration the average information content added by each subsequent RSE-HARQ packet to the received data. From this it may be possible to formulate a mathematical model of the technique and find optimal values for the parameters used to better increase the throughput of the technique.

Chapter 13

Conclusions

This chapter will briefly summarise the theoretical and practical work performed on this project. It will provide an overview of the techniques investigated and how they affect the overall dynamic of the system. Further work is proposed that should allow a system based upon these techniques to be properly implemented and tested. For more particular conclusions and further work based on individual techniques investigated as part of the project please refer to the ends of the relevant chapters.

13.1 Results of theoretical work

The aim of the theoretical work performed during the course of this project was to find techniques suitable for the realisation of a highly asymmetric communications system for the HF band as described in chapter 4. The main problems focussed on for this project were those of combatting poor signal to noise ratio (SNR) (at the receiver, due to the constraints of the system) and compensating for poor channel conditions. As such a considerable amount work has been performed in the area of error correction techniques, using both forward error correction (FEC) and automatic repeat request.

The main metric for assessing the techniques was the power required to implement them at the remote unit. The power budget at the remote unit includes the power required to drive a suitably powerful processor, the power required to implement the on-board electronics and the power required to be fed to the antenna. The latter includes the power radiated from the antenna, power loss associated with using

a poor quality antenna and inefficiencies at the power amplifier from using back-off associated with the peak to average power ratio (PAPR) of the modulation technique. The power transmitted from the antenna has to be sufficient that the SNR at the receiver is of a level to allow reliable decoding of data. Decoding the signal must be reliable enough that few repeat requests have to be generated, thus reducing the length of time that the remote unit is transmitting to transfer a given length of data.

To this end, it was decided that a single tone technique with a low PAPR should be used at the remote unit. BPSK modulation was used as it eased the implementation of other techniques. However this ought to be changed to offset-QPSK, which provides a constant modulus output wave at the remote unit transmitter, in a real system. In principle, this should be relatively simple to implement by modifying the BCJR algorithm in the semi-blind turbo equaliser to allow for more states. However the complexity of the equaliser will be increased and will require more processor cycles (the BCJR algorithm complexity increases by the square of the number of modulation symbols). The decision to use a single-tone modulation technique was also related to considerations regarding the channel equaliser elucidated below. Making this change could allow a highly power efficient transmit power amplifier to be used on the remote unit.

It was decided that low density parity check (LDPC) codes, which are modern forward error correction codes with very good noise performance, should be used to encode data transmitted from the remote unit. These codes require substantial processor resources to decode, which is not an issue as the base station, which will be decoding them, is assumed to have the processing capacity to cater for this. Encoding random LDPC codes also requires large processing and memory resources compared to those available on low-power DSP controllers, however there is a family of algebraically generated LDPC codes known as quasi-cyclic LDPC (QC-LDPC), which allow an encoder to be implemented which requires far less memory and processor cycles. Such an encoder can be implemented on a simple, low power DSP controller and as such the remote unit can be based on such a device.

Compensating for the effects of the channel poses a problem. A channel equaliser is a component of a RF communications receiver which mitigates the effects of the channel on the received signal. The HF channel is poor, with long multipath delays (in the order of milliseconds) and strong frequency-selective fading effects. It is currently popular to compensate for these poor conditions using parallel tone waveforms such as orthogonal frequency division multiplexing (OFDM), which greatly

simplify the implementation of a channel equaliser at the receiver. Unfortunately OFDM is unsuitable for use as a transmit waveform in the battery operated remote unit, as the encoding of such a waveform requires high processing power (generally an inverse FFT is required at the transmitter) and the peak to average power ratio (PAPR) of OFDM is very high compared to single tone techniques, which will reduce the efficiency of the transmit amplifier. The fact that the base station has no specified computational limitations means that a complex equaliser can be used in its receive chain. Additionally, the time taken to decode the information is not critical, with the only requirements imposed from the requirement to send automatic repeat request (ARQ) packets back to the remote unit in the event of a failed packet decode. For these reasons it was decided that a complex channel equaliser should be used to compensate for the channel effects of a single-tone transmit waveform.

The channel equaliser selected for use with the system was based on a new design by Gunther et. al. [87]. It is a semi-blind LDPC turbo decoder, an iterative algorithm which is capable of simultaneously equalising the incoming signal, decoding the LDPC packet used to encode the signal and generate independent channel estimates over the length of the received packet. The equaliser requires a rough channel estimate to be supplied to it which allows it to converge upon the correct channel estimate and reduces the overall error rate of the LDPC decoding process. This technique uses the structure of the LDPC code itself to generate channel estimates based on the received signal and therefore requires very little probe data to be sent along with the packet, when compared to current approaches such as the MIL-STD-188-110B standard waveform. A lower packet overhead means that the remote unit has to spend less time on air to transmit the same amount of data which reduces the energy requirements to transmit a given length of data. This may be especially pronounced given the small amounts of data to be transmitted at any one time. The few kilobytes specified to be sent a day amount to a small number of packets required to be sent, given that the FEC encoder uses the 2044-bit quasi-cyclic LDPC (QC-LDPC) code investigated for use with this project. Where the MIL-STD-188-110B is designed for long, continuous transmission of data, with a large amount of probe data to help a discrete equaliser adapt to the channel before any payload is attempted to be decoded, the new equaliser technique will allow short, opportunistic bursts of a few LDPC packets in length, without any need for long synchronisation patterns to be sent before any given transmission.

Automatic repeat request (ARQ) techniques were also investigated for use with the system. The ARQ techniques investigated were hybrid-ARQ (HARQ) types, which,

again, utilise the structure of the forward error correction used to encode the transmitted data. These techniques have been shown to improve overall throughput of a communications system. A new HARQ technique has been developed and demonstrated, this technique assembles repeat packets generated from linear combinations of bits and has been shown to provide good throughput performance compared to other known techniques, without requiring a high-bandwidth, low error feedback path. This new technique provides a theoretical insight into how the mutual entropy of a transmitted and received data streams may be maximised and is worthy of further investigation. Unfortunately it is not yet known how these HARQ techniques can be integrated to work with the other algorithms explored for use with this project.

13.2 Results of practical work

The practical work performed as part of this project included the design and implementation of a transceiver which was implemented on a custom PCB for the remote unit, and the design and performance evaluation of a buried antenna to provide some basic performance metrics of the poor antenna that will be used with the project.

The remote unit printed circuit board (PCB) was successfully designed and manufactured, and besides a few insignificant design issues the board worked satisfactorily. The unit used a low power 16-bit fixed point digital signal processor (DSP) controller manufactured by Microchip. This unit has only a small amount of data memory (6 kB) and the execution core was clocked at 64 MHz. At this speed the processor was able to generate a 1/2 rate 2044-bit QC-LDPC code in approximately 312 ms, which is short enough that continuous data transmission could take place at speeds of a maximum of about 3 kbps. By using a shorter length or lower rate LDPC code, or by increasing the clock speed it should be possible to improve this throughput at the expense of performance in noise.

A number of buried antennas were characterised to provide information on how a ‘poor quality’ antenna might behave. A radome was designed and manufactured to house the buried antenna and tests were conducted with the antenna located at various depths, with and without the radome. The tests confirmed that the resonant frequencies of the buried antennas were lower down the frequency range than would be expected for a similar length antenna in a proper configuration. These measure-

ments fitted well with the theory of an antenna radiating into a dielectric similar to the ground type. The voltage standing wave ratios of the antennas dropped to approximately 1.1-1.2 at the lowest frequency resonant points, showing that the antennas were radiating well at their resonant frequencies. This surprising result coupled with the lower resonant points of the antennas, suggests that such a configuration might actually work rather well for short-range, near vertical incidence skywave (NVIS) links, which operate at lower frequencies than longer-range systems.

13.3 Recommended further work

This project has opened up many avenues for pursuing further work. This is covered at the end of the corresponding chapters in detail but will be summarised here. The LDPC semi-blind turbo equalisation technique should be better characterised with different LDPC codes to see how changing the rate and codeword length affects performance. Alternative means of generating initial channel estimates should also be examined and the resulting performance improvements evaluated. The semi-blind turbo equalisation technique must be integrated with a good HARQ system, which may be challenging. In particular the novel random sum of elements (RSE-HARQ) technique should be integrated with an equaliser to assess its performance in a more realistic simulated HF channel.

The hardware that was produced was more a proof-of-concept than a finished system. It proved that a low powered PIC microcontroller unit can be used to arbitrate the board resources for a modem system and simultaneously encode long LDPC packets for transmission. It is highly unlikely that this unit could be used to receive and decode transmissions encoded with in a complex FEC format as this operation is typically much more computationally intensive. It would be worth more closely examining the hardware design to improve the performance with a view to creating a prototype radio system that was capable of long range radio transmissions to the base station. In another closely related point, it may be worth examining the GNU Radio platform and universal software radio peripheral hardware and examining how this might be better adapted and improved for use with this system, possibly by creating some bespoke base station hardware based upon it. For both sides of the link it is required that some interface electronics be designed to implement the processing hardware to a real antenna. This may be especially challenging for the remote unit, which may be required to interface with very poor antennas which may cause

problems for most current off-the-shelf hardware.

References

[1] C. Watterson, J. Juroshek, and W. Bensema, “Experimental confirmation of an hf channel model,” *Transactions on Communications Technology, IEEE*, vol. 18, pp. 792–803, Dec 1970.

[2] ITU, “Radio noise,” *ITU-R Recommendation P.372-8*, 2003.

[3] B. Sklar, *Digital Communications Fundamentals and Applications*. Prentice Hall, 2nd ed., 2001.

[4] ITU, “Use of high frequency ionospheric channel simulators,” *ITU-R Recommendation F.520-2*, 1993.

[5] S. Hubbard, J. Peterson, E. Majer, P. Zawislanski, K. Williams, J. Roberts, and F. Wobber, “Estimation of permeable pathways and water content using tomographic radar data,” *The Leading Edge*, November 1997.

[6] ETSI, “Digital radio mondiale (drm); system specification,” October 2005.

[7] K. Davies, *Ionospheric Radio*. Peter Peregrinus Ltd., 1990.

[8] VOACAP, “Voacap quick guide.” Cited from http://www.voacap.com/, 2010.

[9] L. Teters, J. Lloyd, G. Haydon, D. Lucas, and F. Stewart, “Ioncap/voacap model 1983/1993,” 1983.

[10] S. Haykin, *Communication Systems*. Wiley, 4th ed., 2000.

[11] C. Pantjiaros, G. Gott, P. Laycock, M. Broms, and S. Boberg, “The mapping of hf spectral occupancy over northern europe,” *Antennas and propagation*, pp. 4–7, April 1995.

[12] L. Christofi, *Measurement of HF NVIS radio channel parameters with application to the design of very high rate modems.* PhD thesis, University of Manchester, 2001.

[13] W. Furman and J. Nieto, “Understanding hf channel simulator requirements in order to reduce hf modem performance measurement variability,” in *Proc. NORDIC Shortwave Conference Proceedings*, pp. 6.4.1–6.4.13, August 2001.

[14] ITU, “Testing of hf modems with bandwidths of up to about 12 khz using ionospheric channel simulators,” *ITU-R Recommendation F.1487*, 2000.

[15] E. T. LLC., “The usrp product family.” Cited from: http://www.ettus.com/downloads/er_broch_trifold_v5b.pdf.

[16] R. Proesch, *Technical Handbook for Radio Monitoring HF: Edition 2009*. Books on demand GmbH, 2009.

[17] S. Ford, *ARRL's HF Digital Handbook*. ARRL, 2001.

[18] J. Williams, *The illustrated dictionary of mass communications*. Lotus press, 2007.

[19] H. W. Silver, *Ham radio for dummies*. Wiley, 2004.

[20] U. department of defence, “Mil-std-188-110b: Interoperability and performance standards for data modems,” March 2000.

[21] NATO, “Modulation and coding requirements that must be common to ensure interoperability of 2400 bps linear predictive encoded digital speech transmitted over hf radio facilities (stanag 4197),” April 1984.

[22] NATO, “Technical standards for single channel hf radio equipment (stanag 4205),” April 2007.

[23] NATO, “Characteristics of 1200/2400/3600 bits per second single tone modulators/demodulators for hf radio links (stanag 4285),” February 1989.

[24] NATO, “Characteristics of a robust, non hopping, serial tone modulator/demodulator for severely degraded hf radio links (stanag 4415),” February 1998.

[25] NATO, “Technical standards for an automatic radio control system for hr communication links (stanag 4538),” February 2009.

[26] NATO, “Technical standards for non-hopping hf communications waveforms (stanag 4539),” June 2005.

[27] G. Yi and K. B. Letaief, “Space-frequency-time coded ofdm for broadband wireless communications,” in *Proc. 2001 Global Telecommunications Conference*, vol. 1, pp. 519–523, 2001.

[28] J. Wilson, P. Green, and B. Allen, “Meeting with sponsor on 13/11/08.”

[29] Duracell, “Alkaline technical bulletin.” Cited from: [http://www1.duracell.com/oem/Pdf/others/ATB-5.pdf].

[30] P. (press release), “Panasonic launches type 18650 li-ion batteries with a peak capacity of 3.1 ah.” Cited from: [https://industrial.panasonic.com/eu/news/nr201005IE002/nr201005IE002/Press_Release_Li-Ion_NNP_E.pdf].

[31] R. Arland, *ARRL’s Low Power Communication: The Art and Science of Qrp*. ARRL, October 2007.

[32] G. Radio, “Gnu radio official website.” Cited from http://GNUradio.org/redmine/wiki/GNUradio.

[33] P. Cannon, M. Angling, and N. Davies, “Damson hf channel characterisation-a review,” in *Proc. 21st Century Military Communications Conference*, vol. I(2), 2000.

[34] B. Jacobsen, V. Jodalen, P. Cannon, O. Smith, and M. Angling, “Hf radio propagation at high latitudes: Observations and predictions for quiet and disturbed conditions,” *Space Weather Workshop*, December 2001.

[35] H. Haralambous, L. Economout, C. Pantjiaros, and L. Christofi, “Monitoring of hf spectral occupancy in cyprus,” in *Proc. IET 11th International Conference on Ionospheric Radio Systems and Techniques (IRST 2009)*, pp. 42–45, 2009.

[36] D. Percival, M. Kraetzl, and M. Britton, “A model for hf spectral occupancy in central australia,” in *Proc. MILCOM 97*, vol. 1, pp. 346 – 350, Nov 1997.

[37] D. Sumic and R. Vlasic, “An assessment of hf nvis radio system reliability,” in *Proc. 4th International Conference on Ports and Waterways – POWA 2009*, 2009.

[38] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. The University of Illinois Press, 1949.

[39] D. Mackay, “Good error correcting codes based on very sparse matrices,” *IEEE Trans. on Information Theory*, vol. 45, no. 2, pp. 399–431, 1999.

[40] J. Ratzloff and G. Hand, “Users’ guide and reference manual for the voacap and rec533 circuit analysis programs,” 1993. NTIA technical memorandum – 93-157.

[41] ITU, “Method for the prediction and performance of hf circuits,” *ITU-R Recommendation F.533*, 1993.

[42] A. Stocker, M. Muriuki, and E. Warrington, “Comparison of oblique sounding measurements and voacap predictions of a mid-latitude path,” in *Proc. Ionospheric radio Systems and Techniques, 2009. (IRST 2009). The Institution of Engineering and Technology 11th International Conference on*, pp. 65–68, IET, April 2009.

[43] S. Ritchie and F. Honary, “Advances in ionospheric propagation modelling at high latitudes,” in *Proc. Ionospheric radio Systems and Techniques, 2009. (IRST 2009). The Institution of Engineering and Technology 11th International Conference on*, pp. 32–37, IET, April 2009.

[44] M. Walden, “The extraordinary wave mode: neglected in current practical literature for hf nvis communications,” in *Proc. Ionospheric radio Systems and Techniques, 2009. (IRST 2009). The Institution of Engineering and Technology 11th International Conference on*, pp. 27–31, IET, April 2009.

[45] C. Langlais and M. Helard, “Mapping optimisation for turbo-equalisation improved by iterative demapping,” *IET Electronics Letters*, vol. 38, pp. 1365 – 1367, Oct. 2002.

[46] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[47] J. Nieto, “Constant envelope waveforms for use on hf multipath fading channels,” in *Proc. MILCOM ’08*, Nov 2008.

[48] J. Nieto, R. Buckley, and W. Furman, “Waveform and rf power amplifier interdependencies in battery-powered tactical high frequency radio applications,” in *Proc. Ionospheric radio Systems and Techniques, 2009. (IRST 2009).*, pp. 1 –5, apr. 2009.

[49] R. F. et. al., “Power amplifiers and transmitters for rf and microwave,” *IEEE Transactions on Microwave Theory and Technology*, vol. 50, pp. 814–826, Mar 2002.

[50] T. Wilkinson and A. Jones, “Minimisation of the peak to mean envelope power ratio of multicarrier transmission schemes by block coding,” in *Proc. Vehicular Technology Conference, 1995 IEEE 45th*, vol. 2, pp. 825–829 vol.2, Jul 1995.

[51] R. van Nee, “Ofdm codes for peak-to-average power reduction and error correction,” in *Proc. Global Telecommunications Conference, 1996. GLOBECOM ’96. Communications: The Key to Global Prosperity*, vol. 1, pp. 740–744 vol.1, Nov 1996.

[52] M. Friese, “Ofdm signals with low crest-factor,” in *Proc. Global Telecommunications Conference, 1997. GLOBECOM ’97., IEEE*, vol. 1, pp. 290–294 vol.1, Nov 1997.

[53] S. Thompson, A. Ahmed, J. Proakis, and J. Zeidler, “Constant envelope ofdm phase modulation: spectral containment, signal space properties and performance,” in *Proc. Military Communications Conference, 2004. MILCOM 2004. IEEE*, vol. 2, pp. 1129–1135 Vol. 2, Oct.-3 Nov. 2004.

[54] S. Slimane, “Reducing the peak-to-average power ratio of ofdm signals through precoding,” *Vehicular Technology, IEEE Transactions on*, vol. 56, pp. 686–695, March 2007.

[55] P. Robertson and S. Kaiser, “The effects of doppler spreads in ofdm(a) mobile radio systems,” *IEEE Vehicular Technology Conference*, 1999.

[56] D. J. MacKay, *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[57] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimal decoding algorithm.,” *IEEE Transactions on Information Processing*, vol. 13, pp. 260–269, 1967.

[58] R. Tanner, “A recursive approach to low complexity codes,” *Information Theory, IEEE Transactions on*, vol. 27, pp. 533–547, Sep 1981.

[59] R. Gallagher, *Low Density Parity Check Codes*. PhD thesis, M.I.T., 1963.

[60] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd ed., 1992.

[61] D. MacKay and R. Neal, “Near shannon limit performance of low density parity check codes,” *Electronics Letters*, vol. 33, pp. 457–458, Mar 1997.

[62] S. Lin, L. Chen, J. Xu, and I. Djurdjevic, “Near shannon limit quasi-cyclic low-density parity-check codes,” in *Proc. Global Telecommunications Conference, 2003. GLOBECOM ’03. IEEE*, vol. 4, pp. 2030–2035 vol.4, Dec. 2003.

[63] Z. Li, L. Chen, L. Zeng, S. Lin, and W. Fong, “Efficient encoding of quasi-cyclic low-density parity-check codes,” in *Proc. Global Telecommunications Conference, 2005. GLOBECOM ’05. IEEE*, vol. 3, p. 6, Nov.-2 Dec. 2005.

[64] H. Tang, J. Xu, S. Lin, and K. Abdel-Ghaffar, “Codes on finite geometries,” *Information Theory, IEEE Transactions on*, vol. 51, pp. 572–596, Feb. 2005.

[65] Y. Y. Tai, L. Lan, L. Zeng, and K. A. S. Abdel-Ghaffar, “Algebraic construction of quasi-cyclic ldpc codes for the awgn and erasure channels,” *Communications, IEEE Transactions on*, vol. 54, pp. 1765–1774, Oct 2006.

[66] B. Leiner, “Ldpc codes - a brief tutorial.” cited from http://bernh.net/media/download/papers/LDPC.pdf, 2005.

[67] W. Leung, W. Lee, A. Wu, and L. Ping, “Efficient implementation technique of ldpc decoder,” *Electronics Letters*, vol. 37, pp. 1231–1232, Sep 2001.

[68] L. Ping and W. Leung, “Decoding low density parity check codes with finite quantization bits,” *Communications Letters, IEEE*, vol. 4, pp. 62–64, Feb 2000.

[69] S.-Y. Chung, T. Richardson, and R. Urbanke, “Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation,” *Information Theory, IEEE Transactions on*, vol. 47, pp. 657–670, Feb 2001.

[70] M. Luby, “Lt codes,” in *Proc. Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pp. 271–280, 2002.

[71] E. Dahlman, S. Stefan Parkvall, J. Johan Skold, and P. Beming, *3G Evolution - HSPA and LTE for Mobile Broadband*. Academic Press, 2 ed., 2008. pp. 119-123.

[72] Y. Inaba, T. Saito, and T. Ohtsuki, “Wlc16-2: Reliability-based hybrid arq (rb-harq) schemes using low-density parity-check (ldpc) codes,” in *Proc. Global Telecommunications Conference, 2006. GLOBECOM ’06. IEEE*, pp. 1–5, 27 2006-Dec. 1 2006.

[73] Y. Cao, J. Gu, L. Qi, and D. Yang, “Degree distribution based harq for irregular ldpc,” *Electronics Letters*, vol. 42, pp. 363–364, March 2006.

[74] X. Li, Y. Cao, and D. Yang, “An improved degree distribution based harq for ldpc,” in *Proc. Wireless Communications, Networking and Mobile Computing, 2006. WiCOM 2006.International Conference on*, pp. 1–4, Sept. 2006.

[75] S. Sesia, G. Caire, and G. Vivier, “Incremental redundancy hybrid arq schemes based on low-density parity-check codes,” *Communications, IEEE Transactions on*, vol. 52, pp. 1311 – 1321, Aug 2004.

[76] M. Zesong, Z. Miao, and K. Jingming, “Type ii hybrid-arq schemes of ldpc codes based on information-nulling rate-compatible algorithm,” in *Proc. ITS Telecommunications, 2006 6th International Conference on*, pp. 569 – 572, June 2006.

[77] H. Jenkac, T. Mayer, T. Stockhammer, and W. Xu, “Soft decoding of lt-codes for wireless broadcast,” in *Proc. 14th IST Mobile and Wireless Communications Summit*, June 2005.

[78] Z. Ding and Y. Li, *Blind Equalization and identification*. Marcel Dekker, 2001.

[79] R. Casey and T. Karp, “Blind equalization of 8-psk signals aired in the high-frequency band,” in *Proc. Digital Signal Processing Workshop, 12th - Signal Processing Education Workshop, 4th*, pp. 88 – 93, Sept 2006.

[80] S. Haykin, *Adaptive Filter Theory*. Prentice Hall, 4th ed., Sept 2001.

[81] R. Koetter, A. Singer, and M. Tuchler, “Turbo equalization,” *Signal Processing Magazine, IEEE*, vol. 21, pp. 67–80, Jan. 2004.

[82] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *Information Theory, IEEE Transactions on*, vol. 20, pp. 284–287, Mar 1974.

[83] G. K. Kaleh and R. Vallet, “Joint parameter estimation and symbol detection for linear or nonlinear unknown channels,” *Communications, IEEE Transactions on*, vol. 42, pp. 2406–2413, Jul 1994.

[84] R. Lopes and J. Barry, “Blind iterative channel identification and equalization,” in *Proc. Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 7, pp. 2256–2260, 2001.

[85] R. Lopes and J. Barry, “Exploiting error-control coding in blind channel estimation,” in *Proc. Global Telecommunications Conference, 2001. GLOBECOM ’01. IEEE*, vol. 2, pp. 1317–1321, 2001.

[86] J. Gunther, D. Keller, and T. Moon, “A generalized bcjr algorithm and its use in iterative blind channel identification,” *Signal Processing Letters, IEEE*, vol. 14, pp. 661–664, Oct. 2007.

[87] J. Gunther, M. Ankapura, and T. Moon, “A generalized ldpc decoder for blind turbo equalization,” *Signal Processing, IEEE Transactions on*, vol. 53, pp. 3847–3856, Oct. 2005.

[88] R. Straw, L. Cebik, and D. Hallidy, *The ARRL Antenna Book*. ARRL, 21 ed., May 2007.

[89] J. Carr, *Secrets of RF Ciruit Design*. McGraw-Hill Professional, 2000.

[90] J. Heys, *Practical Wire Antennas: Effective HF designs for the radio amateur*. RSGB, 1989.

[91] H. Wheeler, “Useful radiation from an underground antenna,” *Journal of Research of the National Bureau of Standards*, vol. 65D, pp. 89–91, Feb 1961.

[92] R. Fenwick and W. Weeks, “Submerged antenna characteristics,” *IEEE Trans. AP-11*, p. 296, May 1963.

[93] J. Entzminger, T. Treadway, and S. Talbot, “Measured performance of subsurface dipoles,” *RADC-TR-69-221 Technical Report*, June 1969.

[94] H. Bussey and E. Larsen, “Buried antenna performance. development of small resonant buried antennas,” *RADC-TR-74-169 Technical Report*, June 1974.

[95] G. Bingeman, “A simple dummy antenna.” cited from: [http://www.km5kg.com/dummy.htm], 2010.

[96] R. Silberstein, “Subsurface antennas and the amateur,” *The ARRL antenna compendium*, vol. 1, pp. 133–139, 1985.

[97] R. Severns, “Skin depth and wavelength in soil.” cited from: [http://www.antennasbyn6lf.com/files/ground_skin_depth_and_wavelength.pdf].

[98] D. Daniels, *Ground Penetrating Radar*. IET, 2nd ed., 2004.

[99] R. Lewallen, “Eznec demo v. 5.0.41,” 2010.

[100] J. Costas, “Synchronous communications,” *Proceedings of the IRE*, pp. 1713–1718, 1956.

[101] M. Corporation, “dspic language tools libraries (ds51456b),” 2004.

[102] Microchip, “dspic30f6011, dspic30f6012, dspic30f6013, dspic30f6014 data sheet (ds70117e),” 2004.

[103] E. Murphy and C. Slattery, “All about direct digital synthesis,” *Analog Dialogue*, vol. 38-08, pp. 1–5, August 2004.

[104] A. Devices, “Ad9913 dds datasheet rev. a,” 2010.

[105] Minicircuits, “Era 3+ datasheet.”

[106] A. Devices, “Ad5333 datasheet,” 2000.

[107] A. Devices, “Ad8667 datasheet,” 2007.

[108] L. Technologies, “Ltc6911-1/ltc6911-2 datasheet (691112f),” 2004.

[109] W. Hayward, R. Campbell, and B. Larkin, *Experimental Methods in RF Design*. ARRL, 1st (revised) ed., 2009.

[110] Rohde and Schwarz, “Series2000 fk2100/fk2100m antenna tuning unit - data sheet,” 2006.

[111] M. Corporation, “Mplab c30 compiler user’s guide (ds51284c),” 2004.

[112] T. Instruments, “Beagleboard system reference manual rev. c4,” Dec 2009. cited from: http://beagleboard.org/static/BBSRM_latest.pdf.

[113] J. Wilson, P. Green, and B. Allen, “Rapid prototyping with gnu software defined radio,” in *Proc. NORDIC HF Conference 2010*, August 2010.

[114] I. P1901.1, “Standard definitions and concepts for spectrum management and advanced radio technologies (draft),” March 1996.

[115] T. Instruments, “Small form factor software-defined radio development tools (bulletin).” Cited from http://focus.ti.com/lit/ml/sprt406a/sprt406a.pdf, 2007.

[116] R. Leschhorn and B. Buchin, “Military software radios - rohde and schwarz status and perspectives,” tech. rep., Rohde and Schwarz, 2004.

[117] Aerostream, “Aerostream official website.” Cited from http://www.commradio.com/.

[118] Lyrtech, “Lyrtech official website.” Cited from http://www.lyrtech.com/, 2010.

[119] J. Buchanan, “Windrm website.” Cited from http://n1su.com/windrm/.

[120] K. Tan, J. Zhang, and J. Fang, “Sora: High perfomance software radio using general purpose multi-core processors,” *6th USENIX Symposium on Networked System Design and Implementation*, p. 75, 2009.

[121] F. Systems, “Flexradio systems website.” Cited from http://www.flexradio.com/.

[122] C. Bayona, “Gnu radio official website.” Cited from http://www.softrockradio.org.

[123] P. S. Foundation, “Python (official website).” www.python.org.

[124] E. T. LLC., “Usrp datasheet.” Cited from http://www.ettus.com/downloads/ettus_ds_usrp_v7.pdf.

[125] A. Devices, “Ad9860/ad9862 datasheet,” 2002.

[126] J. Proakis and D. Manolakis, *Digital Signal Processing, Principles, Algoritms and Applications (3rd Ed.)*. Prentis Hall, 1996.

[127] J. Wilson, P. Green, and B. Allen, “An asymmetric low-power hf modem link,” in *Proc. NORDIC HF Conference 2010*, August 2010.

[128] D. Mackay, “David mackay’s website: [www.inference.phy.cam.ac.uk/mackay/],” 2009 September.