A wireless sensor network for in-crop sensing

A thesis submitted to The University of Manchester for the degree of
PhD
in the Faculty of Engineering and Physical Sciences

2013

Peter Michael Green

School of Electrical and Electronic Engineering

Contents

Total word count 61090

List of figures

List of tables

Glossary

ADC Analog to digital converter.

ADS A software package from Agilent for design and simulation of radio frequency circuits.

balun a device or circuit used to convert between balanced and unbalanced signals.

BPSK Binary phase shift keying: A modulation technique where information is encoded by switching the phase of a carrier between two different values.

DMM Digital multi meter: A device for digitally measuring voltage and current.

drop-out voltage The minimum voltage between input and output for a regulator to operate correctly.

DSSS Direct sequence spread spectrum: A method for spreading signal across a wider spectrum by combining it with a chipping code.

FHSS Frequency hopping spread spectrum: A method for spreading signal across a wider spectrum by repeatedly changing operating frequency.

GRP Glass reinforced polyester.

HFSS Radio frequency 3D electro-magnetics simulation software from Ansys (formerly Ansoft). Originally the name stood for high frequency structure simulator.

ISM Industrial, scientific and medical: These frequency bands were originally allocated for non-communication industrial, scientific and medical uses, they are also used for low power license exempt communication.

little-endian Data stored with least significant byte first.

LNA Low noise amplifier: An amplifier designed to amplifier small signals while adding as little noise as possible. Typically used as the first stage in a receiver.

node a device on a network.

O-QPSK Offset quadrature phase shift keying: a modulation scheme where two signals are phase modulated onto in-phase and quadrature carriers with the transitions of the modulation of the two carriers being offset.

PA Power amplifier: An amplifier designed to increase the power level of signals before transmission.

PCB Printed circuit board.

PIC A microcontroller from Microchip's PIC range of microcontrollers.

RC filter A filter that consists of a resistor and a capacitor.

RF Radio frequency.

SFTP Secure file transfer protocol. A protocol for securely transferring files based on the SSH remote access protocol.

supernode A node that forwards data in a hierarchical network. May also be referred to as a repeater.

VNA Vector network analyer: a test instrument used to measure the magnitude and phase of signal transmission and reflection through a radio frequency system.

Zigbee A networking protocol created by the Zigbee alliance based on IEEE 802.15.4 with hierarchical and mesh routing features.

Abstract

University: The University of Manchester

Author: Peter Michael Green

Degree: PhD

Title: A wireless sensor network for in-crop sensing

Date: 9th December 2013

This project looks at the operation of wireless sensor nodes in rapeseed using antennas located close to the ground, through both measurements of the propagation loss with test equipment and through building and operating an experimental wireless sensor network. Rapeseed is a technically challenging crop in which to operate a wireless sensor network, due to its dense foliage from ground level up to the canopy.

To facilitate these activities two measurement systems have been devised and constructed. One measurement system was designed to allow the measurement of propagation loss in and around the crop. The other was designed to allow the power profile of sensor nodes to be measured. Both of these systems provided information to guide the design of the experimental wireless sensor network. The measurements with the propagation measurement system along with the received signal strength and message loss results from the deployment provide information on the distances over which communication is likely to be feasible with antennas located close to the ground.

Sensor nodes and a base station to collect the data from them were designed based on Microchip's PIC18F45J20 microcontroller, MRF24J40 transceiver chip and MiWi software stack with customisations for power management and to record the received signal strength of packets arriving at the base station.

Initial propagation loss measurements in a rapeseed crop were made at Rothamsted Experimental Station in 2011. Further propagation loss measurements and a deployment of the sensor nodes in a rapeseed crop were made at Tatton Dale Farm in 2012.

Declaration and Copyright

No portion of the work referred to in the 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.

i The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see http://www.manchester.ac.uk/library/aboutus/regulations) and in The University's policy on Presentation of Theses.

Some source code included on the optical disc provided to the examiners is based on third party code and has not been cleared for public release. As such this optical disc should not be made available to the general public.

The "Environmental monitoring in grain" paper included in Appendix B is ©2010 IEEE. Reprinted, with permission, from Graham Parkinson, Environmental monitoring in grain, Instrumentation and Measurement Technology Conference (I2MTC), May 2010.

Acknowledgements

The author would like to thank the following people for the assistance they have provided.

The supervisors Mr. Peter R. Green and Dr. Danielle George for all the support and advice they have provided through the project.

Prof. Jon West from Rothamsted Research for allowing us to perform experiments there.

Mr. Richard Reeeves for providing the location for our deployment and experiments at Tatton Dale Farm.

The mechanical workshop in the School of Electrical and Electronic Engineering at the University for assisting in the design and construction of enclosures, mounts and similar, to support the practical work.

About the author

In 2009 the author received a first class MEng with industrial experience in Electronic Systems Engineering from The University of Manchester. Projects undertaken during this degree included developing an interface to connect a multi-channel charge sensitive amplifier to a FPGA system and developing improved interface hardware for use in the control systems undergraduate laboratory.

The author has also been involved in the development of open source software. In collaboration with Bas Steendijk the author developed an event driven pascal network library called lcore. The author also contributed many bugfixes to the Debian Linux distribution and since May 2012 has become a Debian Developer with upload and voting rights.

In early 2012 the author collaborated with Mike Thompson to start a project called Raspbian which aimed to recompile all software in Debian for the Raspberry Pi and potentially other devices that have ARMv6 processors with VFPv2 floating point units. The project was a success and was adopted by the Raspberry Pi Foundation as their preferred distribution in July 2012.

Chapter 1

Introduction

There is currently pressure to reduce usage of water [13] and chemicals [121] in agriculture. By measuring conditions on a more local basis (how local depending on the particular crop and condition being monitored) watering, fertilisation and pesticide applications can be targeted where and when they are needed instead of being used in a blanket fashion across the whole field. This use of more localised data is known as precision agriculture[12].

A wireless sensor network (WSN) is a wireless network designed to collect data from sensors and transfer it to a place where it can be logged, forwarded over a longer distance network (such as a mobile phone network) and/or used to control actuators [20]. A minimal wireless sensor network consists of end nodes which perform the sensing and a base station (also called a gateway) to collect the data. Depending on the network design, the data may be sent either directly or indirectly to the base station. If it is sent indirectly, dedicated repeater nodes may be used or the regular nodes may perform a repeater function. Each approach

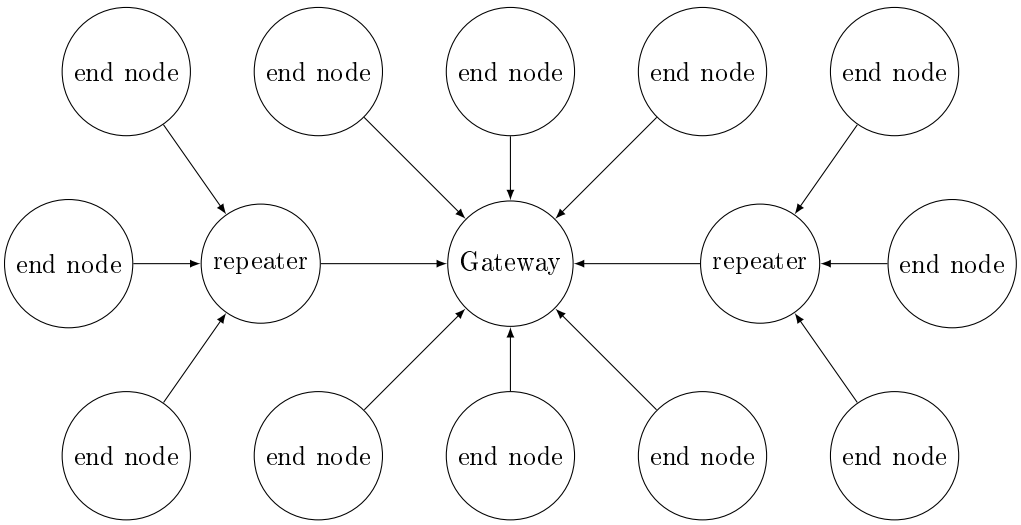


Figure 1.1: An example of a hierarchical network

has pros and cons and the best one will depend on the specific application. Figure 1.1 shows an example of one of the possible topologies for a wireless sensor network.

Wireless sensor networks have potential as a means of gathering the information required for precision agriculture. However, there are a number of significant challenges to creating a wireless sensor network that is viable to deploy and operate in a crop field.

Since mains power is unlikely to be available and maintenance is likely to be difficult, the nodes in a wireless sensor network (especially the end nodes which are most numerous) must normally operate from battery supplies for long periods. Power scavenging methods may also be considered for some networks but are very situation-specific. The requirement to run off batteries for a long period of time requires the average current drawn from the batteries to be kept to very low levels. For example, if it is desired to run off a set of AA cells for one and a half years, the average current drawn must be of the order of 100 µA. Also, wireless sensor networks may have to operate in conditions where propagation losses are very high. A block diagram of a typical sensor node can be seen in Figure 1.2

It is important that water is kept away from the electronics inside a sensor since it can both affect the operation of the electronics and cause corrosion. Since farming generally takes place outdoors, the nodes must be sealed to keep out water. Furthermore due to the large variations in temperature a node that is sealed with normal room air inside is likely to suffer from condensation. To avoid this, the air or other gas sealed inside the node must be very dry. This can be achieved through the use of a desiccant or by filling the node with dry gas during assembly. The need to ensure the node is well sealed and filled with dry gas makes battery replacement even more inconvenient than it would be in an indoor network. Therefore, long battery life is even more important. However, fields are generally ploughed at the end of each season which is likely to mean the nodes will have to be removed and re-deployed. So a battery life of one growing season would likely be acceptable but more would be desirable.

A significant problem is developing an antenna system that provides a sufficiently good channel[1] for reliable communication while also not interfering with conventional farming operations or being unduly expensive. In particular, antennas near the ground are likely to give very poor channel performance due to both reflections from the ground and to the attenuation from the radio path passing through the crop.

Antennas near or above the crop canopy are likely to give much lower channel attenuation

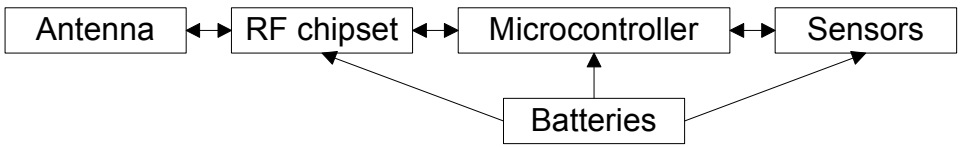


Figure 1.2: Block diagram of a typical sensor node

[1]Some authors regard the antennas as being part of the channel while others regard the antennas as being separate from the channel. The author believes that the former approach makes sense in applications where there is plenty of free space around the antenna but that in a crowded environment the latter approach makes more sense as the effects of the antenna cannot be captured by a gain figure alone.

but will be more expensive due to the cost of constructing supports, and have the potential to interfere with farming equipment such as spray booms. High gain antennas can reduce the channel loss but if they cannot be made and deployed cheaply they will not be cost effective for a wireless sensor network. Most deployments so far have used simple antennas, either wire monopoles or commercial "rubber duck" omnidirectional antennas [57][123][17]. Measurements of the channel conditions with a proposed antenna type and location in the crop, are crucial to gauging the feasibility of using that antenna arrangement for a deployment.

The cost of nodes must also be kept low, otherwise the cost of the network may become greater than the benefit that can be gained from the data collected. Exactly how low the cost per node must be kept will depend on both the number of nodes required to provide useful information and the size of the benefit to be gained from the data collected. In order to give some rough perspective the economic margin of winter oilseed rape to farmers is typically £510 per hectare [121] and it is reasonable to assume that benefits from a wireless sensor network will be a fraction of this.

1.1 Aims and motivations

The primary focus of this project has been to explore the operation of 2.4 GHz wireless sensor networks in a rapeseed oil crop. Exploration has been performed through both measurements of the RF propagation conditions with a channel measurement system built using RF test equipment and through designing, fabricating and operating a demonstration wireless sensor network through a growing season. Rapeseed was suggested as a crop to work in by Jon West, a plant pathologist at Rothamsted Research, who works with crops, including rapeseed [126].

Rapeseed was attractive for this research for a number of reasons; firstly it is widely grown in the UK, with 705 thousand hectares grown in 2011 [30], and was therefore available for us to work in. Secondly it a challenging crop to work in since it has dense and irregular foliage from ground level right up to the canopy of the crop (unlike some crops where there is a relatively open space between ground and canopy). Finally, it is vulnerable to pests including insect pests from the Meligethes, Phyllotreta, Psylliodes, Ceutorhynchus and Asineura genuses [56], and fungi including Blumeria graminis, Puccinia striiformis, Pyrenopeziza brassicae, Rhynchosporium secalis, Leptosphaeria maculans and Mycosphaerella graminicola [18]. In particular Leptosphaeria maculans infection is sensitive to moisture and therefore information on moisture in the field could potentially help target treatments.

On the other hand rapeseed is only a moderate value crop and therefore getting a commercial benefit from a wireless sensor network may not be practical. While cost considerations will be mentioned in this thesis, the focus is on whether operating a network is technically feasible, not whether doing so is economically viable for a commercial crop. Even if a sensor network is not viable in a commercial crop, it may still have value as a tool for assisting plant researchers in monitoring their crop trials.

The work with rapeseed has taken two main strands. One strand was to design, develop and use a RF propagation measurement system to determine what the propagation losses in

rapeseed actually were. The other strand was to design, construct, program and operate an experimental wireless sensor network in a rapeseed crop on a farm. As part of developing the sensor nodes for the experimental wireless sensor network, a system for characterising their power consumption profile was also designed and constructed. Both radio paths through the crop near the ground and radio paths near just above the crop canopy have been investigated.

The network operations and measurements were performed in the 2.4 GHz ISM band. It had been hoped to take measurements in other bands too but due to the difficulty of organising visits to access the crop, this did not happen. The 2.4 GHz band was chosen as it was perceived by the author at the time to be the most common wireless band with the most options available for RF transceiver integrated circuits (ICs). For the experimental wireless sensor network, hardware and software from Microchip Technology were used, specifically the MRF24J40 [68] and the PIC18F45j11 [72]. These were chosen because of the author's previous familiarity with Microchip products and the perceived need to get the system working quickly so it could be deployed as soon as possible. The combination of the Texas Instruments MSP430F22x4 microcontroller [114] and CC2500 transceiver [111] was also trialled initially but was rejected due to software problems. The power demands of the Microchip and Texas Instruments combinations are broadly comparable as is the radio transmit power. The CC2500 has the advantage of offering a low modulation rate high sensitivity mode while the Microchip device has the advantage of using DSSS which the author believes is the most appropriate modulation technique for short transmissions in environments where significant multipath effects may be present.

Two strategies for antenna location were explored. One strategy was to transmit near the ground; this keeps the antennas out of the way of spray booms, though the nodes would still have to be removed before harvest, but this leads to a poor radio channel. The other strategy was locating antennas near (either just above or just below) the canopy of the crop.

It was initially planned to perform the deployment and measurement sessions at Rothamsted Research and some initial measurement sessions were performed there. However, the distance involved made working there somewhat impractical and a co-operative local farmer (Richard Reeves, Tatton Dale Farm) was found, allowing the experiments and a trial to be performed more locally. However arranging trips to perform maintenance on the experimental wireless sensor network and make RF propagation measurements remained problematic. It had been hoped that deployments could be made in two different years with the second deployment learning from the mistakes of the first, but difficulties gaining access to the crop meant this did not happen.

Separately, the use of RF itself as a sensing mechanism has also been briefly explored. Initial simulation work on this was done during the first year of the PhD. It was planned to continue this with a laboratory based test rig in the third year but there were delays due to equipment failures and the work is still incomplete. Some initial work on investigating different antenna types for wireless sensor networks was also performed in the first year but did not lead to any new discoveries or indications. Finally, work was carried out in the first year to develop boards for the WINES 2 Wireless Sensor Networked For Industrial Processes project [27] which is detailed in appendix A

1.2 Structure of the thesis

The thesis is structured as follows.

- Chapter 2: This chapter contains a review of the literature focusing on wireless sensor networks in agriculture, antennas for wireless sensor networks, wireless propagation in foliage and existing methods of field propagation measurement.
- Chapter 3: This chapter goes through many of the considerations facing wireless sensor nodes. These considerations include dealing with the limitations imposed by battery supplies, the choice of what topology to use and whether that topology should be static or dynamic, what frequency bands are available, the implications of antenna type and location and the modulation types available.
- Chapter 4: This chapter covers the early investigation carried out on possible antenna types for wireless sensor networks.
- Chapter 5: This chapter describes the system designed for measuring power consumption profiles of wireless sensor nodes with high dynamic range.
- Chapter 6: This chapter describes the measurement system developed for measuring wireless propagation loss in the field. It also describes the variant of that system used to measure the gain and harmonic production of power amplifiers.
- Chapter 7: This chapter contains information on the evaluation of sensor node hardware platforms and the development of sensor nodes based on those platforms for use in demonstration wireless sensor networks.
- Chapter 8: This chapter describes the work done in preparation for the main field activities including early measurements and a deployment at the University to test hardware.
- Chapter 9: This chapter describes the fieldwork performed at Tatton Dale Farm and its results. The work included both the operation of an experimental sensor network and the making of measurements with the propagation measurement system. This chapter describes all the field work carried out for the project and the results it produced including both measurement sessions with the field propagation measurement system and operation of the demonstration wireless sensor network.
- Chapter 10: This chapter documents the conclusions of the project and contains suggestions of how the work may be taken forward in the future.
- Appendix A: This appendix contains details of the side work undertaken for the WINES 2 WSN4IP project including development of a battery board, a sensor board and a bulk charging station.
- Appendix B: This appendix contains a copy of a research paper that partially resulted from the work the author performed for WINES 2 WSN4IP.
- Appendix C: This appendix documents the investigations into the possibility of using RF as a sensing mechanism.

- Appendix D: This appendix documents a four port resistive radio frequency splitter that was designed and constructed to support the sensor node development work.
- Appendix E: This appendix describes the material on the optical disc supplied to the examiners.
- Appendices F to M: Contain source code developed for the project.
- Appendices N to S: Schematics for various boards designed as part of the project.

Chapter 2

Literature review

This chapter looks at the literature surrounding wireless sensor networks in precision agriculture. It begins by outlining the motivations. Next it introduces the concept of precision agriculture and looks briefly at options other than wireless sensor networks for gathering information for precision agriculture. Next it looks at a sample of existing wireless sensor network deployments in agriculture. Finally it discusses specific aspects of operating wireless sensor networks, some specific to the agricultural setting and some more general.

2.1 Motivations

Farmers apply pesticides and fertilizers to their crops to protect them from pests and increase yields. These chemicals often end up contaminating surface and ground waters. Contamination with pesticides may have toxic effects on aquatic plants and animals [21] and may require the installation of treatment plants to meet drinking water quality standards [15]. Nutrients from fertilizers can cause excessive plant growth in water bodies, a process called "cultural eutrophication" [104].

As well as the active ingredients themselves, impurities in agricultural chemicals can be an issue. For example, Rohana Chandrajith et al report the introduction of significant quantities of radionuclides into rice paddies in Sri Lanka due to use of poor quality phosphate fertilizer [22].

According to the Pesticide Action Network [36], pesticide run off into English rivers is a "big problem for farmers". They assert that due to contamination in rivers caused by run off there is a danger that many more pesticides will be banned.

In 2006 the European Commission announced "A Thematic Strategy on the Sustainable Use of Pesticides" [23]. They state that despite previous efforts unwanted amounts of certain pesticides can still be found in soil and water, and that residues exceeding regulatory limits still occur in agricultural produce. Among the goals of the strategy are "to reduce the levels of harmful active substances" and "to encourage low-input or pesticide-free cultivation". In 2009 the EU passed Directive 2009/128/EC to implement the strategy.

R.J. Hillocks from the Natural Resources Institute stated in 2011 that EU policy had resulted in the loss from the UK market of some important crop protection products and was likely to result in the loss of more [44]. In 2008 the environmental consultancy ADAS investigated four possible proposals from the European Commission and Parliament for controlling pesticide use [1]. They concluded that under the most stringent of these proposals the "Parliament Substitution" proposal there could be a 77 % loss in production of brassicas in the UK.

Agricultural water usage is a problem in drier parts of the world. For example 30 % of the groundwater used for irrigation in the US comes from the High Plains Aquifer [107]. However the High Plains Aquifer is being drained considerably faster than it is being recharged. As a result the amount of water available for agriculture in those areas will decline as the aquifer is drained.

According to Postel [86] while many countries are depleting aquifers on a regional basis, five countries are depleting groundwater reserves faster than they can be replenished on a national scale. These countries are Saudi Arabia, Libya, Egypt, Pakistan and Iran. Saudi Arabia in particular irrigated the desert on a massive scale. Agricultural water usage there more than tripled between 1980 and 2006 and the country became self-sufficient in wheat. However in 2008 aquifer depletion and rising costs of pumping from ever greater depths lead the Saudis to announce the phaseout of irrigated wheat production.

There is also high demand for food in the world and this along with other factors has caused high food prices in recent years. In 2008 Trostle from the US department of agriculture investigated food price rises [120]. He noted that production growth was slowing but determined this was a slowly changing factor and likely played a negligible role in the recent price rises. On the demand side he noted that rising per-capita incomes in developing countries had not only increased their consumption of staple foods but also caused them to diversify their diets to include meat, dairy products and vegetable oils. This in turn amplified the demand for grains and oilseeds. Prices of fertilisers, fuel and pesticides rose due to the rise in the price of crude oil. There were also short term factors involved such as poor weather conditions in some countries in 2006 and 2007 and political factors such as countries increasing their buffers against supply shortages and increasing the cost of exporting. According to the World Bank food prices dropped after 2008 but then rose again to reach an all time peak in August 2012. Since then prices have fallen slightly but in July 2013 prices were still about double their levels in 2005 [127].

2.2 Precision agriculture

The combination of environmental pressures and a need for higher yields poses a problem. How to increase the amount of food (agricultural output) produced while reducing the amount of water and/or chemicals (agricultural inputs) used in producing it. One possibility is precision agriculture, that is rather than taking decisions based on averages for a large area, local variations are recognised and the application of agricultural inputs is adjusted based on those variations [96].

To take local variations into account requires information on what those variations are. Many methods of gathering such information on an automatic or semi-automatic basis have

been proposed in the literature.

One option is to gather information from above through either airborne or satellite imagery [129]. Unmanned aerial vehicles have also been proposed for this use [131]. Such a system can quickly cover a large area, however while it provides information on variations in the current state of the crop, the underlying conditions causing those variations must be inferred indirectly. Cost considerations will also limit temporal resolution.

Another option is to mount sensors on mobile farm equipment. In 2004 D. Ehlert et al built a variable rate fertilisation system based on a pendulum sensor measuring crop density [33]. When the crop was more dense more fertilisation was applied. Commercial tractor mounted sensors are now available for measuring the nitrogen condition through leaf or canopy spectral reflectance [118]. It is also possible to include sensors on harvesting equipment to monitor yield. For example Won Suk Lee et al proposed a yield monitoring system for silage [119]. In their system a moisture sensor in the silage chopper was connected via Bluetooth to a monitoring system in the silage trailer. The monitoring system in the trailer also had load sensors to measure the weight of the material in the trailer.

Sensors mounted on farm equipment have obvious advantages, one sensor can cover a large area, supplying power is unlikely to be a problem and the system is just a minor alteration to an activity the farmer already performs. However this also brings a key disadvantage, sensors on farm equipment only provide data when the equipment is passing through the field. Therefore sensors on farm equipment can be used to vary the rate at which a product is applied but would not be much use for determining when to apply it.

Another option is data loggers. A data logger can be left in the field measuring one or more conditions at defined time intervals. It can then be manually read out. However the need for someone to manually read out the data results in a time delay in the availability of recordings and according to Huma Zia et al [133] "The expense of data loggers, infield reliability, calibration requirements and the need to extract data from them individually, prevent them from being used for a dense and long term deployment of sensors.".

Another option is wired sensor networks. Jin Yulong and Yang Jiaqiang from Zhejiang University proposed a RS485 based system for greenhouse control [50]. Luis Ruiz-Garcia et al claim that wired networks are very reliable and stable but that required cabling engineering is very costly [96]. The author feels that while wired systems may be feasible in a tightly controlled environment like a greenhouse or possibly a vineyard they would be impractical to deploy in a regular farm field.

Another option is standalone sensor nodes each with a direct connection to a cellular phone network [54]. Such a system has the obvious advantage over a wireless sensor network that nodes can easily be deployed at low densities as there is no need for them to communicate with each other. However for high node densities the cost of such a system is likely to be higher than a wireless sensor network for several reasons. Firstly cellular modules are more expensive than short range wireless modules ($50 each versus $16 each in small volume). Secondly wake-up times are likely to be longer as the cellular module must have time to associate with the mobile network before sending data increasing power use and required battery size. Thirdly every node must have an antenna whose type and location provide sufficient cellular signal to allow communication with the cellular network. This may be

difficult in areas with poor cellular coverage. Finally there will be user fees payable for each device using the cellular phone network and the author believes these are likely to be greater with multiple independent cellular connections than with a wireless sensor network that aggregates the data and passes it through a single cellular connection.

A final option is to operate a wireless sensor network [96], that is a network which is dedicated to collecting data from wireless sensors.

2.3 Wireless sensor networks in agriculture

The author has not found any evidence of other groups working to deploy wireless sensor networks in rapeseed. Many authors have tried to operate such networks in agricultural situations. A number of examples are discussed below.

One of these projects is LOFAR-Agro, a side project of the LOFAR project [63]. The aim was to assist in the control of Phytophtora in potato crops through monitoring of humidity and temperature. Baggio published a conference poster [14] in 2005, claiming they were in the process of setting up their system but giving few details. In 2006 Langendoen published a paper titled "Murphy Loves Potatoes" [57], giving more details of the project and reporting that the project had failed to meet its objectives due to a large number of hardware and software issues.

Their system used the CC1000 radio transceiver from Chipcon (now TI) operating in the 868 MHz SRD band. The nodes were mounted on poles at a height of 75 cm to keep the antennas above the crop. There was no discussion of the antenna type in the text but from the images it appears to be a simple wire monopole emerging from the top of the node casing. At the edge of the field was a system they referred to as the "field gateway" which collected the data from the nodes and was connected to the internet via an existing Wi-Fi network. The field gateway had a 5 m tall antenna with the intention of being able to communicate directly with most nodes in the field. Multi-hop routing was implemented in case some nodes could not directly reach the field gateway. There were 100 nodes in the deployment. The distance between the gateway and the nodes was not discussed. The individual nodes were powered by internal batteries while the gateway was powered by a combination of a large battery with solar panels. A wireless reprogramming system was also included to allow the nodes to be updated without having to physically open them.

They encountered a number of issues while operating the system. There was a poor interaction between between power management in the microcontrollers and timer use in TinyOS. Different TinyOS modules made different assumptions about how the transceiver hardware should be configured leading to incompatibilities. There were problems with the routing layer when large numbers of nodes were used which had not shown up with smaller configurations in the laboratory. There were battery failures due to unexpectedly high power consumption. Finally they had problems with poor software management practices leading to the wrong version of the software being installed on the nodes. Overall only 2 % of the messages sent by the nodes were received.

After these experiences they decided to simplify the system [53]. In particular multi-hop support and wireless updating were removed. Single hop communication was achieved over

distances of 40 m and 51 % of the messages sent were received. However there were still problems, in particular the solar panel used to power the gateway was insufficient to keep the battery charged leading to the system going down every night.

A group from the Commonwealth Scientific and Industrial Research Organisation (CSIRO) in Australia published a paper in 2007 about wireless sensor networks for agriculture [123], but focusing on pasture for grazing of animals rather than fields of crops. Earlier revisions of their system operated in the 433 MHz ISM band while later revisions operated in the the 915 MHz ISM band. They measured soil moisture with commercial capacitance based sensors and also planned to monitor the visual condition of the pasture with a camera based system. In addition to this, they also used mobile sensor nodes attached to the cows to monitor the cows' behaviour. The fixed nodes were well above the ground (an exact height was not given), spread out and in open sunlight. This allowed them to avoid the battery life problem by using solar power to run their nodes unlike wireless sensor nodes among fields of crops which may be low to the ground and often under a crop canopy. The mobile nodes were mounted on the necks of the cows with the antennas flat along the top of the collar. Initially they had placed the antennas sticking up from the collar but they discovered when they did this the cows quickly destroyed the antennas.

The paper did not contain any network performance information for the fixed sensor nodes but did discuss packet loss versus distance for the nodes mounted on the cattle. Interestingly 40 % of the packets were successfully received even at a distance of 200 m.

The group from CSIRO published a later paper in 2011 giving more details of their system [24] and charting its history across multiple deployments starting with the cattle monitoring deployment described above. According to this paper the cattle monitoring deployment described above used an in-house developed self organising time division multiple access scheme which achieved multi-hop communication through flooding messages to nodes. However they found that this flooding methodology was not scalable, particularly when the cameras were introduced into the system.

In 2004 a group from Intel Research, Cassia Technology and Pacific Agri-business Research Centre/AgCanada published details of their experiences operating a wireless sensor network to measure temperatures in a vineyard [17]. The goal of measuring these temperature variations was twofold, firstly to allow selection of the most appropriate variety of wine grapes for each location and secondly to allow steps to be taken to protect the grapes from frost. Rather than develop their own sensor nodes, this project used a node known as the Berkeley Mica II mote [26], and its associated sensor board, though they did modify the power supply and sensor arrangements. They mostly avoided the battery life issue by the brute force method of using six D-size alkaline cells in every node. The system operated at 915 MHz. Once again the system was TinyOS based. The system performed multi-hop transmission and each data packet was resent five times to allow for packet loss. They reported that the radio performance in the field was significantly lower than in the laboratory and that there were a few days where the network was unstable and had to be reset. No evidence was found of the project continuing into further years.

Susan A. O'Shaughnessy and Steven R. Evett from the USDA-ARS Conservation and Production Laboratory investigated two different wireless sensor arrangements for automatic

irrigation with a centre pivot irrigation system [82]. The XBee-Pro was used for communication in the 2.4 GHz ISM band with a whip antenna. The nodes were powered with a combination of a solar panel and a lead acid battery. Infra-red thermometers were used to measure the temperature of the crop canopy as an indication of crop stress. One system used sensors mounted on adjustable platforms in the field and single hop networking. The other used sensors mounted on the irrigation system and was tested with both single hop and mesh networking firmware. They found that even with the mesh networking firmware, the nodes mounted on the irrigation system performed substantially worse than those on platforms in the field and attributed this to the steel framework of the pivot arm and towers.

A group from the University of Maryland and Carnegie Mellon University developed a system for monitoring conditions in greenhouses [60]. One interesting aspect of their system is that they can support multiple sensors per node and even actuators attached to the nodes. The paper stated that they were using multi-hop self organising communication but gave no details of the software or hardware used to achieve this, nor any statistics on how it performed. The same group also operated a wireless sensor network in a tree farm using the EM50R "wireless data logger" from Decagon Devices [59]. The EM50R is a commercial product that operates in the 2.4 GHz ISM band, again no data on network performance was given. They published a later paper in 2011 [58] stating they had implemented a six node wireless sensor network for monitoring substrate moisture and electrical conductivity in the root zone but again gave no details on network performance.

2.4 Antennas for wireless sensor networks

It is noticeable that in all of the projects cited above that provide information on the antennas, they are using relatively simple antennas. The antennas tend to be either wire monopoles or commercial "rubber duck" antennas. The internal structures of rubber duck antennas varies with the desired frequency range and physical size. According to Martin Pot a typical 2.4 GHz rubber duck antenna consists of a folded dipole while rubber duck antennas for lower frequencies such as those used by CB radio (around 27 MHz) are often helical antennas due to size constraints [87]. Investigating commercially available 868 MHz rubber duck antennas showed a mixture of quarter wave monopoles and half wave dipoles. Physically larger and higher gain rubber duck antennas are also available commercially and the author believes they likely use some form of collinear structure [85] to achieve this gain.

Monopoles and dipoles are simple and cheap and when mounted vertically are omnidirectional in the horizontal plane allowing communication with nodes in multiple directions using the same antenna. A half wave dipole has a doughnut shaped radiation pattern and when mounted vertically has its peak gain in the horizontal direction. However the peak gain for an ideal half wave dipole is only 2.1 dB and a significant amount of energy is transmitted at elevations above and below the horizontal [35]. Ideally a quarter wave monopole would be operated over an infinite perfect ground plane and in this case its radiation pattern would be the top half of that of a dipole, with no energy delivered below the horizontal and a peak gain 3 dB greater than a dipole. However in practice such perfect ground planes do not exist and the radiation pattern of a monopole will vary depending on grounding conditions

with the peak gain being typically at an angle above the horizontal [134]. Assuming that the ground is flat and the antenna is vertical, energy that is not sent horizontally is either wasted or worse reflected off the ground, where it may interfere with the original signal.

There are two main options for improved antennas. One option would be a high gain omnidirectional antenna that emits a larger proportion of the radio energy horizontally and a smaller proportion at angles away from the horizontal. Another would be a directional antenna. Omnidirectional antennas are easy to deploy and allow for communication in multiple directions, but the maximum gain available is constrained by the one dimensional scaling. Directional antennas can produce higher gains as the energy is focused in two dimensions. Another option may be to use more than one directional antenna; this could make certain types of communication and measurement easier and may also allow for direction finding but would add the problem of handling the signals to and from the different antennas. This could be done either through splitting the antenna feed, switching between the antennas or using multiple radio transceivers. However each of these options comes at a cost. Using a RF splitter will reduce signal levels and could create nulls in the response due to destructive interference between the antennas. Switching between antennas will add cost to the node and also add operational complexity since the node can only listen for communications in one direction at a time. Using multiple RF transceivers will add to both the cost and power consumption of the node. Antennas that can be electronically switched between multiple directions are also a possibility but bring similar operational complexities to switching between multiple antennas.

The main impact of improved channel performance from higher gain antennas is likely to be an increase in the distance that can reasonably be covered in a single radio hop and hence a reduction in the number of hops needed to get a message to its destination. For a network with pre-determined repeater nodes this may mean that a smaller number of such nodes is required. For a network where the number of nodes is determined by the radio conditions rather than by agricultural needs it may reduce the total number of nodes in the network. Reductions in transmit power do not have a significant impact on node power usage as a large proportion of node power is spent on waiting for incoming transmissions [52]. The use of directional antennas may also reduce packet collisions and therefore lower retransmit rates, but this is unlikely to be significant at the low data rates involved.

In 2009 Nilsson published a paper on the use of antennas for direction finding in wireless sensor networks [79]. The options investigated include an “adcock pair" which uses the difference between signals from pairs of dipoles, a “pseudo doppler” antenna which uses rapid switching between a set of dipoles, and an “electronically switched parasitic element antenna" where parasitic elements are switched between grounded and isolated to make them act as either “reflectors" or “directors" or the reactance between the parasitic elements is controlled in a more gradual manner, by biasing a varicap diode. After describing these options the paper goes on to give details of a low cost “electronically switched parasitic element antenna" designed for use in the 2.4 GHz ISM band. However in the prototype no switching was implemented. The reflector elements were instead permanently connected and the directing element was permanently isolated. The paper states that that a real version of the antenna for use in a sensor node will need these replacing with electronic switches but does not go into details on how they plan to do this. Their simulation and test results also

seem to be based on perfect connection and perfect isolation, ideals that may be difficult to achieve electronically.

Dunlop and Cortes from the Department of Electronic and Electrical Engineering at the University of Strathclyde have done simulation based research on the effect of introducing antennas with switchable direction to wireless sensor networks [31]. They appear to realise (though they do not explicitly state) that transmit power reductions will bring little power savings due to the power characteristics of low power radio nodes. The main improvement they found came from the reduced interference and hence lower retransmit rates that could be achieved.

A group from Tyndall National Institute in Ireland have been working on the development of antenna clusters to give a good approximation of a spherical radiation pattern in an underwater application. They did this by using three antennas and a switching system [81]. They do not seem to give any details of the speed of switching or how the signals were combined.

Hobbyists are another possible source of designs. Many people build their own 2.4 GHz antennas for Wi-Fi (which like many wireless sensor networks usually operates in the 2.4 GHz ISM band). However relatively few of their designs are appropriate for wireless sensor network use, being either too expensive or too directional. An interesting antenna design was described by Martin Pot [88]. The claimed gain of 6 dBi is better than a monopole and the simple bent wire construction could likely be fabricated at an acceptable cost. The antenna is a little large to put on every sensor node but may be an option for repeater nodes in a hierarchical network or for the base station that collects data from the network. Later Pot added more elements to the antenna and claimed that the new version of the antenna had a gain of about 8 dBi.

There are many commercial antennas on the market with varying levels of gain and directionality but generally the sellers provide little to no information on their internal structure. Higher gain models also tend to be relatively expensive which is undesirable for wireless sensor nodes. They may however be useful for repeater nodes in a hierarchical network or for the base station that collects data from the network.

Integrating the antenna onto the printed circuit boards of the node is another option; the advantage of this method is that it avoids most of the costs normally involved in manufacturing an antenna. The disadvantage is that it restricts the antenna's size and the planar nature could make it difficult to achieve vertically symmetrical radiation patterns. It also generally means that the antenna must be internal to the node which is likely to be a disadvantage. For example, if the node is buried in the ground a wire antenna above the ground is likely to perform much better than a PCB antenna at ground level.

2.5 Wireless propagation in and around foliage

The LOFAR-Agro project performed measurements of propagation loss in potato fields [115]. They made the measurements using the receive signal strength indication feature in their Mica2dot node hardware, using a nominal transmit power of +10 dBm which was the maximum supported by their node hardware. Their measurements show a radical change in

performance over the season with the average RSSI at 23 m changing from -86 dBm in July to -115 dBm in August[1]. This shows the vital importance of taking multiple measurements over a season in order to obtain a realistic picture of the circumstances under which a network must operate.

Hubner et al. [45] give some results from 2.4 GHz propagation measurements in a vineyard. Unlike most other projects they tested at a variety of heights and their results show substantial improvements in signal strength as the height is increased from 10 cm to 110 cm, and again to 210 cm. While a vineyard is a somewhat different environment from a field of rapeseed their results clearly show that it is important to gather data at different heights so an appropriate compromise between radio propagation and deployment practicality can be found.

Mestre et al. [66] investigated a number of existing models for attenuation of radio signals by vegetation and compared those models to their own measurements in the 2.4 GHz ISM band in rosemary, escallonia and creeping juniper. They attempted to take measurements at 1 m, 2 m, 3 m, 4 m, 5 m and 10 m. However for rosemary and creeping juniper the attenuation at 10 m was too high to measure with their equipment configuration. The 2 m and 10 m values for escallonia in their results also appear suspiciously low compared to the surrounding measurements.

The models they investigated were the Weissberger MED (Modified Exponential Decay) model [125], the COST235 model, the fitted ITU-R model, the Early ITU-R model and the Single Vegetative Obstructive model [48]. These models all suggest that loss can be predicted by a formula of the form $L = af^b d^c$ where L is loss in dB, f is frequency (in GHz), d is distance in vegetation (in metres) and a b and c are constants of the model[2].

After comparing the results from the models with the results from their tests they stated that none of the models were able to correctly predict the attenuation values. They then went on to suggest that the models could be adapted by multiplying them by a constant for the vegetation type. They determined the constant by using the least squares fitting procedure. They then presented correlation factors between the predicted and measured values. Since their measurements were taken at a single frequency in the 2.4 GHz band the frequency dependence in the models was not analysed. Therefore their calculations were simply comparing the choice of the exponent c applied to distance by the different models.

The exponent of 1 used by the Weissberger MED and Single Vegetative Obstructive models fitted best for rosemary. The exponent of around 0.25 used by the COST-235 model and the fitted ITU-T model worked best for escallonia but as mentioned above the escallonia results look suspicious. All models provided the same level of correlation in creeping juniper [3]. This would seem to indicate that the exponent of 1 was probably the best of those tested.

Ngandu et al tested wireless propagation near ground level in a variety of vegetation around the University of Johannasberg campus which they selected as being similar to crop foliage [78]. The foliage types tested were ivy, long dense grass, shrub and a hedge. They used the received signal strength functionality of XBee radio modules with a nominal transmit

[1] figures were read from the graph in their report

[2] The Single Vegetative Obstructive model has no frequency dependence and the Weissberger MED changes constants at a distance beyond that considered in their tests

[3] The author suspects this may be an error in the paper

power of 0 dBm to measure the signal and set a cut-off threshold of -75 dBm for acceptable performance. A transmitter was placed near the edge of the foliage and a receiver more deeply inside. In the ivy they tested at distances of 3 m and 5.4 m, and found acceptable results in both cases, it is not clear why they did not try longer distances. They also tested placing the receiver node on a 15 cm high box and found this improved the signal. In the shrub the signal was around their threshold at a distance of 1.6 m and below it at a distance of 2.4 m. In the grass the signal was above their threshold at 1 m but below it at 2 m. In the hedge the signal was above their threshold at 2 m, around their threshold at 2.4 m and 3 m and so low they could barely detect it at all at 4 m.

QinetiQ developed "A Generic Model of 1-60 GHz Radio Propagation through Vegetation" for the UK Radiocommunications Agency [89]. While the title of the model talks about vegetation in general, the document focuses on trees. The model was based on measurements made at 12 locations with eight species of tree. The model is aimed at operators of cellular networks. The model splits propagation into three components, a component diffracted around the vegetation, a component reflected off the ground (possibly also going through some of the vegetation) and a component going directly through the vegetation. It then provides methods for calculating each component and combining the results. To calculate the loss through the vegetation it uses an equation based on radiative energy transfer theory. To use this equation requires information on the distance in vegetation, the beam width of the receiving antenna and four parameters that are dependent on vegetation type and frequency. The report tabulates values for these parameters for a number of species of tree and documents experimental and numerical procedures for determining them for other species of tree.

The prevailing strategy among those deploying wireless sensor networks in agricultural situations appears to be to transmit over the crop rather than within it. Hui Liu et al suggest that the antenna should be mounted sufficiently high that with the crop fully grown no more than 40 % of the first Freznel Zone should be obstructed [62].

Zhuohui Zhang from the University of Illinois tested the maximum range of a Bluetooth system at various heights over bare soil, soybean and corn fields [132]. The hardware was described as a "CrossNetTM Wireless CN1100 Node Series" and a "Toshiba Bluetooth PC Card". The transmitter and receiver were placed at the same height on two platforms. Interestingly he found that for each crop there was an "optimal" height and that raising the antennas beyond this height caused the maximum distance to fall. Unfortunately he did not state what the maximum signal loss the radio hardware he was using could tolerate before failing and did not provide any RSSI readings. As a result of this it is difficult to apply his distance figures to other equipment.

Hebel et al from the University of Illinois performed tests over an open grass field using the RSSI measurements from XBee Pro 2.4 GHz transceivers and came to a similar observation that higher antenna locations were not always better [42]. In particular they found that at distances of 50 m and 75 m antennas at a height of 1.5 m gave a greater signal strength than antennas at a height of 2 m. They later concluded [43] that this was due to reflections causing constructive interference in the first Fresnel zone but destructive interference in the second Fresnel zone and suggested a freeware tool called Radio Mobile [25] for calculating losses taking account of Fresnel zone interference for antennas 0.25 m or more off the ground.

They compared the results of this model with their measurements and observed the same dip in signal level in both.

Rao et al investigated propagation loss in a forest environment in India [92]. They used a 1 m antenna height and tested at 915 MHz and 2.4 GHz in three types of forest over distances of 5 m to 20 m. To make the measurements they used an Agilent N5182A to send a BPSK signal to an Agilent N9010A signal analyser [92]. A forest environment is obviously very different from a crop on a farm but the author found the similarity in the results for the two frequency bands to be interesting with at most around 10 dB difference between the figures for the two frequency bands.

Rao et al also investigated propagation at 2.4 GHz in a variety of crop and garden settings [91]. They tested a total of five environments, a corn field, a rice paddy, a groundnut field, a coconut garden and an open grass field. They used an antenna height of 1 m in the corn field and the coconut garden, a height of 15 cm in the paddy and groundnut fields and a height of 2 cm in the open grass field. It is also not clear to what point on the antennas these heights were measured, nor is it clear how the heights were selected. Measurements were performed both with the signal generator and analyser set-up used for the forest measurements and with XBee Pro wireless modules at distances of 5 m to 20 m.

During this literature review no evidence has been found of any projects measuring wireless propagation in or around rapeseed crops either as part of a sensor network or otherwise.

2.6 Effect of rain and moisture on propagation of signals

Hebel et al investigated increase in signal loss due to rain [42]. They made measurements over a distance of 200 m with an antenna height of 2 m and compared results in dry weather with results in heavy rain. Interestingly the results drastically changed across the 2.4 GHz band. For channels 1-15 (representing centre frequencies of 2.410 GHz to 2.440 GHz) the average reduction in signal level was about 5 dB and there was little variation. However for the upper channels in the band the average reduction in signal level increased to about 10 dB and there was also a substantial increase in variation. This suggests that for outdoor applications of the 2.4 GHz ISM band, the lower half of the band should be used if possible.

In initial tests working in a potato crop at 433 MHz the Lofar Agro project reported better propagation when relative humidity was higher [41]. However in later tests at 868 MHz they reported the opposite with higher packet loss on days with high relative humidity [53].

2.7 Monopole antennas and ground planes

A monopole is an antenna formed from a single wire. Their simplicity to construct and omnidirectional nature makes them a popular choice. Ideally the base of a monopole would start from an infinite perfectly conducting ground plane. In this case the radiation pattern would be half that of a dipole. This would give maximum gain immediately above the horizontal and zero gain below the horizontal [35]. For a quarter wave monopole the gain would be approximately 5 dBi.

However, in practice infinite ground planes made of perfect conductors do not exist. Monopole antennas are operated either with finite sized solid ground planes made from good conductors, finite sized ground planes made up of radial wires, the surface of the earth or no deliberate ground plane at all. Zivkovic et al studied the behaviour of monopoles above finite sized circular and square ground planes using both analytical and simulation methods [134]. As the ground plane is enlarged they found angle of maximum directivity converges towards the horizontal but even for a ground plane disc 20 wavelengths in size it is still only 80° from the vertical . For a ground plane half a wavelength in size they found that the maximum directivity had dropped to around 2.5 dBi and the angle of maximum directivity was around 50° from the vertical. They also noted that for all the ground plane sizes considered the horizontal directivity was approximately 5 dB below the maximum directivity. In an agricultural situation the author feels that even a half wavelength ground plane will considerably increase the size of the antenna assembly increasing the risk of damage.

Janek and Evans used the electromagnetics simulation software HFSS to simulate monopole antennas raised above two ground planes [49]. One made of a perfect conductor and one made of a material with a dielectric constant and conductivity similar to the ground's surface in their field location [4]. They claim that they first simulated the monopole in free space and when they did so found it showed characteristics similar to an ideal dipole. They then simulated it at a selection of heights above the two ground planes. They claimed that one of their figures shows for the antenna closest to earth ground "a slight parabolic radiation pattern (shape) emanates from the bottom, leaving space under the parabola where no transmission occurs". Unfortunately they did not give details on their simulation model (in particular how the monopole was fed) and the results were presented only in the form of 3D plots with no scale.

2.8 Propagation measurements in the field

There are several ways of measuring propagation characteristics of an RF channel. One option is to use the receive signal strength indication features in the RF transceivers built into nodes. Among the projects investigated in the previous sections of this chapter this seems to be by far the dominant approach. This has the advantage of being available at no extra hardware cost if a network is already deployed or to be deployed and at low cost even if one is not. It also allows measurement over long periods in a realistic setting. However the RF front ends of a typical sensor node RF chip set are not designed to be precision measurement devices and this will limit accuracy. For example the CC2420 has a specified accuracy of ±6 dB. It is also limited to one frequency band and to signals which are sufficiently strong for the transceiver to receive.

Another method, as described by Zaghloul et al. [130], is to use a network analyser. This method has the advantage of being able to retrieve phase information as well as magnitude information and can be used for multiple frequency bands with only the antennas being changed. However the noise floor of network analysers is usually relatively high (for example the Anritsu MS2036a [11] has a noise floor of approximately -80 dB) requiring the use of a

[4] The author observes that this will likely vary with the weather, the paper does not make it clear whether the numbers used represent dry conditions, wet conditions or average conditions

separate amplifier and/or relatively high gain antennas (rather than antennas representative of those used on the nodes). Furthermore, as a result of the coaxial cables between the network analyser and the antennas, the transmit and receive antenna share a common ground. This is not representative of the conditions that would be found in a sensor network. The author worried that a common ground between transmit and receive antennas may provide a signal path from transmitter to receiver that bypassed the channel under test.

Another method is to use a chirp sounder as described by Sana Salous [97]. This generates a short pulse of frequency modulated signal which is received by a special receiver. Chirp sounding provides a rapid means of measurement and can measure multipath effects but it requires special equipment to generate and receive the chirps.

Thipparaju Rama Rao et al used an Agilent N5182A to send a BPSK signal to an Agilent N9010A signal analyser [92]. There did not appear to be any automatic control of the generator and it is not clear to the author why they chose a BPSK signal over a simple carrier.

Andrew Seville used signal generators and a spectrum analyser under computer control to measure radio signal loss between the inside and outside of buildings with the transmit equipment located inside the building and the receive equipment located in a land rover with a telescopic antenna mast [100]. He used two different signal generators to cover a wide frequency range from 400 MHz to 18 GHz. To control the devices at the two locations a pair of GPIB extenders with a coaxial cable in between was used. Use of a pair of radio modems for control was considered to reduce the cabling requirements but was not used because of poor reliability.

2.9 Measurement of sensor node power consumption

There are two main approaches to measuring the power consumption of a wireless sensor node.

One option is to measure the current consumption (with a known supply voltage) as the node goes through its normal operational cycle. Average and peak values can then be calculated from this measurement. This approach has the problem that it is difficult to achieve sufficient dynamic range. Texas Instruments describe using an oscilloscope with a current sense resistor to make such measurements [75] but also mention the problem of dynamic range and do not propose a solution to it.

Agilent have created the N6781A source/measurement unit for their N6700 series modular power system [2]. This module is specifically aimed at battery drain analysis. Agilent claim that their hardware can provide "the complete current waveform you were not able to see before, from nA to A, in one pass and one picture". They have a video [3] showing measurement of a device that varies between 125 µA and 120 mA using this system. However at around £8000 for the system (with a N6781A module and a N6705B chassis) it is an expensive option for a device that will probably only be used occasionally.

The other possibility is to set up the node to run in a fixed state and then measure current using a multimeter [10]. By repeating this for each state of node operation and then com-

bining the measurements with knowledge of the time spent in each mode, an average power consumption can be calculated. This can avoid the dynamic range issues as each state can be measured using different ranges on the multimeter, but great care is needed to ensure that the time in each state is representative of the node's actual behaviour.

2.10 Power consumption and duty cycles

Low power RF transceivers typically use 10 mA to 20 mA in receive mode [110][68] at approximately 3.3 V. Supplying this current continuously is possible but will require either a large battery or a solar charging system like the one used by CSIRO [123]. Therefore it is highly desirable to design the system in such a way that the radio transceivers spend most of the time in a low power sleep mode.

This raises a problem, if a packet is sent to a receiver that is in sleep mode then it will be lost. The Zigbee standard was designed around the assumption[5] that routing nodes would have a constant power supply [109]. In this case the end device can wake up periodically, transmit data to the router, receive any responses and return to sleep mode. Such operation is acceptable in an environment where mains power is available but in an agricultural situation it is likely that routing nodes will need to be located where mains power is not easily available.

Clearly an approach is needed that allows all nodes to spend some proportion of the time in sleep mode. There are both high and low level approaches to this. At a high level it is possible to have the network wake up periodically for a synchronised "time window". For example the LOFAR-agro project had their nodes wake up for 65 ms every 6.1 s [57]. Such an approach requires a mechanism for introducing new or out of sync nodes to the network. The LOFAR-agro project achieved this by having the node stay awake for 300 ms. This high level approach has the advantage of being simple and predictable but it is also limiting, communications can only take place during the times the network designer chooses to have the system awake and total throughput is limited.

Lower level approaches are also possible. One approach is known as "low power listening" [19]. In a simple implementation the receiver briefly wakes up at regular time intervals and checks for a preamble. The transmitter sends a preamble that is longer than the wake-up interval of the receiver. If the receiver wakes up and detects a preamble it stays awake, waits for it to complete and receives the data packet. This approach has several issues. Firstly the long preambles waste channel capacity and has the potential to block other users. Secondly the receiver must wait for the long preamble to complete before the data can be received. Thirdly all receivers that detect the preamble must remain awake until the addressing information is sent.

A protocol called X-MAC [19] improves on this. Instead of sending a single long preamble the sender transmits a series of short preambles each containing addressing information. When a receiver detects a preamble addressed to it an "early ACK" is sent by the receiver to notify the transmitter that it is ready to receive the data. The transmitter then sends the payload. This allows the data to be received sooner and allows receivers other than the

[5] The author has found mention of low power routers in the latest version of the Zigbee specification but could not find any details of their behaviour.

intended receiver to return to sleep mode more quickly. Pablo Suarez et al made a practical implementation of X-MAC [109] and concluded that it could reduce power usage by up to 90 % over a traditional Zigbee system.

A concern the author sees with these low-level approaches is predictability. If there is an unexpectedly large amount of traffic (for example from defects in higher level code) then the nodes may stay awake continuously and drain their batteries.

Chapter 3

Wireless sensor network design considerations

There are a number of factors that must be considered in the design of a wireless sensor network. These factors include network topology, antenna design and location, power management and cost [1]. These factors are interrelated, for example the choice of topology will impact on the protocol used which will in turn impact on the power requirements. The choice of topology will also impact on which nodes require a viable radio channel to each other which will impact on selection of antenna type and location. Antennas must be selected and located to provide a channel of sufficiently low loss between nodes while keeping disruption to farming practices to a minimum.

3.1 Topologies

At minimum a wireless sensor network consists of a number of sensor nodes and a "base station" (also referred to as a gateway). The sensor nodes (also known as end nodes) collect measurements from internal or external sensors and transmit it via the network to the base station. The base station receives the data from the network and stores it, processes it to make decisions (such as whether to turn on a sprinkler system) and/or relays it over a longer distance network. There may also be nodes that serve to repeat data on the network.

There are three main topologies to consider. The first is a star network where every node transmits directly to the base station as shown in Figure 3.1. This is the simplest type of network but requires every node to have a viable radio channel to the base station all the time which is not always practical.

The second possibility is a mesh network, shown in Figure 3.2. In this topology all nodes other than the base station can play an equal role in the network and nodes communicate with each other to forward packets of data towards the base station along whatever route they judge is best. Since the data path can flow through any combination of nodes, mesh

[1] Both the direct costs of buying, installing and removing the wireless sensor network hardware and the indirect costs of any disruption caused to existing farming practices.

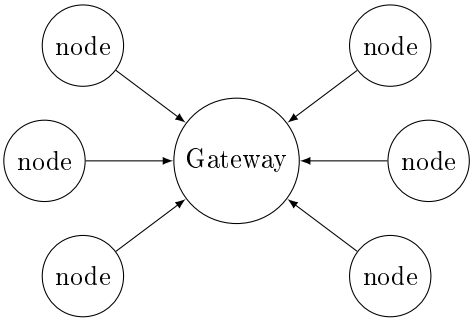


Figure 3.1: An example of a star network

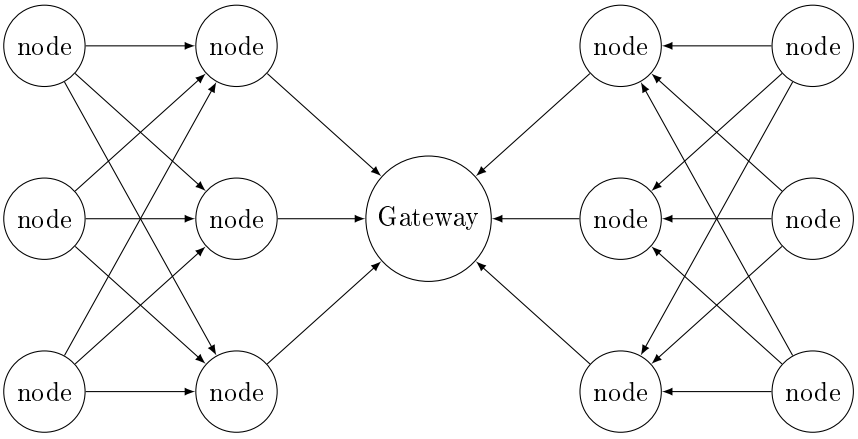


Figure 3.2: An example of a mesh network

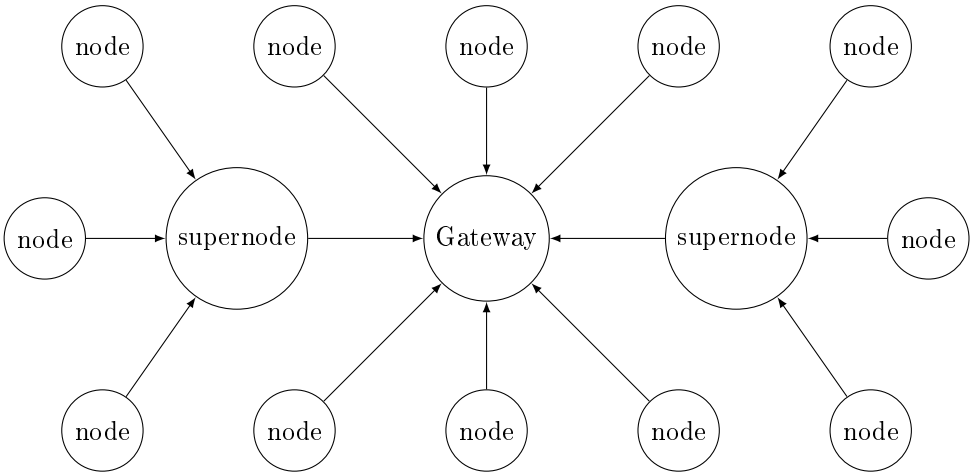


Figure 3.3: An example of a hierarchical network

networks in principle give the highest probability of successfully passing the data from the source node to the base station but often have complex and difficult to predict behaviour. They also create problems for power management. A node that spends most of the time asleep will be difficult to send messages to, and depending on how power is managed, nodes on common forwarding paths may end up using up more power than nodes that merely handle their own data.

The final option for a wireless sensor network deployment is a hierarchical network as shown in Figure 3.3. In a hierarchical network end nodes communicate with supernodes (also referred to as repeaters) which in turn either communicate directly with the gateway or form a mesh or hierarchy of supernodes.

Which nodes in a hierarchical network are supernodes can either be planned in advance or decided dynamically by the nodes themselves. Dynamic decisions allow the nodes to optimise for propagation conditions and to share the power burden of being supernodes. However if the supernodes are chosen in advance they can be fitted with extra hardware to assist them in the role. Such hardware may include larger batteries, antennas with higher gain and/or elevation and range extender ICs which increase the power and sensitivity of the nodes radio system.

After reading about the LOFAR agro project's poor experiences with mesh routing [57] the author was discouraged from mesh networks and concluded that hierarchical networks were probably the most appropriate option for a wireless network. They have predictable behaviour, whilst allowing for multiple paths to be provided from end node to base station and allowing distances greater than could easily be covered by a single radio link.

Towards the end of the project the author has realised that the large scale of a typical farm means that a practical network will likely need to cover distances of hundreds of metres or possibly even kilometres and believes this will rule out simple one-hop and two-hop designs. The author still thinks there is merit in having some nodes be pure end nodes but that a substantial fraction of nodes will have to have repeater functionality.

The author has also realised that redundancy is important in network design. Equipment located outdoors is vulnerable to failures and radio links can unexpectedly become non-viable due to changes in conditions. As a result a network that relies on every node and link continuing to work correctly is likely to be unreliable.

3.2 Addressing and routing

Nodes in a network are identified by addresses. Sometimes a node may have more than one address, for example there may be a permanent address that identifies the node consistently over time and a temporary address that (partially or completely) identifies its location in the network.

One simple approach to addressing and routing is to assign addresses in a hierarchical manner so that a nodes address describes the complete path between it and the gateway[2].

[2]Strictly speaking the "root node" of the network does not have to be the gateway but the author feels this is the most sensible arrangement.

This makes routing very simple, a repeater node can simply compare the destination address with its own address and based on that comparison it can send the packet up or down the hierarchy. Routing loops are also avoided. However this approach has problems, firstly since the path must be represented in the address, the address must be large enough to accommodate this. Secondly if the path from a repeater to the gateway changes, then all nodes behind that repeater must receive new addresses. Microchip's MiWi protocol uses this approach [67] but limits the number of wireless hops between the root node and any end device to two. Zigbee has the option of a similar topology but allows the network designer to select the compromise between the number of levels in the hierarchy and the number of nodes behind each repeater [83].

Another simple approach is flooding, this is where a node sends a packet to all neighbours who in turn forward it to all their neighbours. This will eventually broadcast the packet to every node in the network but it is a very wasteful and precautions must be taken to prevent packets looping forever [90].

Another option is to build routing tables. To create an entry in the routing table a routing request is broadcast to all nodes in the network, replies to the routing request are then used to fill in the routing tables. Care must be taken with this approach to avoid creating routing loops as the strength of links changes. In larger networks the level of broadcast traffic will also be high. Zigbee offers a mesh mode which uses a version of this technique [80].

A wide variety of more complex routing techniques specific to wireless sensor networks have been proposed [5]. The author has not evaluated these in detail.

3.3 Powering sensor nodes from batteries

Power consumption is a major concern in wireless sensor networks. To avoid costly and difficult charging or battery changing the batteries in the nodes need to last for a long time. The required lifetime will depend on the application, but in agricultural applications is likely to be at least one growing season. This, along with the ever present need to keep per node cost down, means that average current drawn from the battery needs to be well under a milliamp which in turn means nodes must spend most of their time in a sleep state.

3.3.1 Average current

To calculate the average current available to a node it is necessary to know the capacity of the batteries and the required battery lifetime. Consider a node powered by a set of Duracell Procell MN1500 AA cells connected in series with a required life of $1\frac{1}{2}$ years. These batteries have a rated capacity of 2700 mAh but this rated capacity is based on a cut-off voltage of 0.8 V and is based on a 75 ohm constant load [32], a very different load to the ones being considered here.

Work by Floyd [37] has shown that batteries tend to provide approximately the nominal capacity stated by the manufacturer when subjected to pulsed load patterns with an average current of around 1 mA. This test scenario is closer to the behaviour exhibited by a wireless

sensor node, than the high current constant draw scenarios represented by the manufacturer's figures. However Floyd still discharged the batteries far quicker than would be acceptable for a wireless sensor node. Floyd's work also revealed that temperature is a major factor in battery life. Sustained freezing conditions were determined to have a very large negative impact but unfortunately he did not test the impact of short periods of freezing conditions in an otherwise normal temperature cycle.

To estimate the acceptable average current for a node some assumptions must be made. For the purposes of these calculations it will be assumed that when operating outdoors the AA battery will only provide half its rated capacity or 1350 mAh. Further it will be assumed that the node must last one and a half years and that there are 366 days in a year giving a total required lifetime of 13176 hours. This puts the maximum permissible average current at just over 100 µA. Obviously larger batteries could increase the current available but this will increase the cost and size of the node.

The current drawn by available RF transceiver ICs with their receivers active is much greater than the average current calculated above. For example the MRF24J40 [68] in receive mode has a rated current consumption of 19 mA. Therefore the nodes must spend the majority of the time in a sleep state where the receiver is turned off. The proportion of the time which the receiver of an end node can be turned on and listening for incoming transmissions is likely to be well under 1 %. It is also necessary to carefully check the current consumption of any other components that will remain powered while the node is in sleep mode. For implementations without range extenders the current consumption during transmission is typically only slightly greater than the current during reception (23 mA for the MRF24J40 [68]). For implementations with range extenders the current during transmission will be significantly higher than the current during receive, for example the CC2591 range extender consumes 3.4 mA in receive mode and 112 mA in transmit mode [112]. However due to the short transmissions in wireless sensor networks this is not a significant component of the overall power budget.

The current consumption of a node in the sensor network is related to the topology of the network. A device that only acts as an end node only needs to wake up for long enough to transmit a packet and receive a reply. On the other hand a node that is providing repeater or base station functionality must remain awake for longer to allow for clock drift between nodes and to give new nodes that have not yet synchronised their clocks a reasonable chance of joining the network successfully. With a star network or a fixed hierarchy where the role of nodes is planned in advance, the nodes providing repeater functionality and the base station can be given larger batteries to allow them to stay awake for longer. However it is still not practical for repeaters running on battery power to remain awake all the time. With a mesh network or a dynamic hierarchy such pre-planning is not possible since it is not known which nodes will be performing repeater duties.

3.3.2 Voltage regulation

The voltage delivered by batteries reduces considerably as they discharge. Therefore the node must be able to operate over a range of battery voltages. The reduction in battery voltage that can be tolerated before the system fails is known as the depth of discharge.

Ideally a node would be able to operate over a wide range of battery voltages, so the batteries could be discharged as deeply as possible. However efficiency and quiescent current considerations must also be considered in designing the power circuitry. There are essentially three options for supplying power from the batteries in a sensor node to the electronics.

The simplest option is to use the batteries to power the microcontroller and transceiver directly without regulation. In addition to the simplicity, this option has the advantage that no power is wasted in regulators. However in a system that does not use a regulator the voltage delivered to the electronics will drop substantially over the life of the batteries. An alkaline AA cell has a nominal voltage of 1.5 V and new cells are typically slightly above this so a device must be able to tolerate a voltage of at least 1.6 V per cell. Duracell consider a battery to be completely discharged when its voltage falls to 0.8 V [32]. For two AA cells in series this would mean that ideally a device would operate over the range 1.6 V to 3.2 V. In practice RF transceivers typically have a narrower voltage range (for example the MRF24J40 is only rated to operate from 2.4 V to 3.6 V). This will limit the acceptable depth of discharge and hence the proportion of the energy in the batteries that can be used. Furthermore while the transceivers are often specified to operate over a wide voltage range, their performance is often only specified for operation at a single voltage. Reduction in voltage is likely to reduce the transmit power and possibly other performance characteristics of the transceiver. This reduction in performance may lead to a situation where links that were initially viable fail during the the deployment.

The second option is to use a linear regulator. The regulator should have both a very low drop-out voltage to allow the batteries to be discharged as deeply as possible and should have a very low quiescent current to minimise power draw when the node is in standby. Linear regulators with very low drop-out voltage and sufficiently quiescent current are available off-the-shelf, for example the LM2936 from Texas Instruments which has a maximum drop-out voltage of 0.4V and a maximum quiescent current under light load of 20 µA. Unfortunately the data sheet quiescent current specifications are for relatively high input voltages, but a small set of measurements using a DMM revealed the quiescent current remained stable as the input voltage was lowered. The downside of using a linear regulator is that more cells must be connected in series to maintain a sufficiently high input voltage for the regulator over the life of the batteries and when the batteries are new a large proportion of the energy taken from the batteries is dissipated as heat in the regulator.

The final option is a switched mode converter. Switched mode converters can provide efficient conversion from the wide range of voltages produced by a battery over its lifetime, to a stable voltage for the electronics. However they require considerably more external components and can be sensitive to PCB layout issues, making them more complex to integrate into a design successfully than the other options. Conventional switched mode converters also tend to have relatively high quiescent currents due to the switching process. However there are now switched mode converter ICs on the market such as the MAX16904 [65] which achieve acceptably low quiescent current (25 µA typical for the MAX16904) through operating in a discontinuous mode, in which they transfer power from the input to the output capacitors in occasional short bursts when under light load.

It may also be possible to extend the battery life of a node by bypassing the regulator when the voltage drops below a predetermined level. However if a suitably low drop-out regulator

was used this is unlikely to result in significant life extension.

3.3.3 Rechargeable versus primary cells

The ability to recharge the cells making up the battery in a node is clearly an advantage since it avoids the need to throw them away when they become discharged. Furthermore if the battery can be charged while it is still in the node it may eliminate the need to remove batteries at all in the node's lifetime and hence allow the node to be completely sealed at production time and never opened by the customer. Avoiding the need for the enclosure to be opened and re-sealed would significantly simplify enclosure design.

Rechargeable batteries also have disadvantages. They are more expensive to purchase and were conventionally regarded as unsuitable for long duration low current applications due to self discharge. However in recent years new "stay charged" rechargeable batteries have been introduced which have much lower self discharge than previous rechargeable batteries. For example the Sanyo eneloop batteries which the manufacturer claims retain 70 % of their charge after 5 years of storage [98].

3.4 Frequency bands

Generally wireless sensor networks are operated in the ISM bands or other license exempt bands to avoid the need for licensing and because of the ready availability of transceivers for these bands. This reduces costs, but also brings the risk of interference both from other communications and possibly from non communication uses of RF.

There are a number of available license exempt bands [64]. Low frequencies are problematic due to low bandwidth and antenna size issues. For example, in the 40.7 MHz ISM band only 40 kHz of bandwidth is available. While the average data rates from a wireless sensor node are of the order of tens to hundreds of bytes per hour, a wider channel bandwidth has two advantages. Firstly a larger bandwidth allows the use of spread spectrum modulation to reduce the effects of frequency selective fading. Secondly it reduces transmission times thereby saving power. Furthermore in the 40.7 MHz band a simple quarter wave monopole antenna would be over a metre long. It is possible to make small antennas for use at low frequencies, for example by using helical designs but such antennas have relatively poor performance. When planning the practical work these considerations lead the author to conclude that low frequency bands were a poor choice, however the author has since been informed that these downsides may be outweighed by reductions in foliage attenuation and it would be worthwhile to investigate channel performance in these bands within a crop environment.

This leaves the 433 MHz ISM band, the 2.4 GHz ISM band and the 5.8 GHz ISM band as the main options for wireless sensor networks. The 868 MHz SRD band and 915 MHz ISM band are also a possibility but typically only one of these bands is available in a given country meaning that two versions of the system will be needed if it is to be sold worldwide. In general, higher frequencies have more bandwidth (though bandwidth at 2.4 GHz and 5.8 GHz is comparable) and smaller antennas for a given gain, while lower frequency bands

typically have better propagation. For example, with free space propagation the fraction of transmitted energy received by the receiver is proportional to the square of wavelength. On the other hand the plane-earth propagation model gives a result that is independent of wavelength. Neither model perfectly reflects real-world propagation on earth.

The most popular license exempt band is the 2.4GHz band. It is used for a wide variety of systems including Wi-Fi for local area networks, Bluetooth for communication between portable devices, ZigBee and similar protocols for low data rate networks and a wide variety of proprietary radio products. This popularity brings both advantages and disadvantages. It means that there is a wide choice of economical transceivers but it also means the chance of interference from another system is greater. It also has other potential problems such as being in the water absorption band [6].

In urban areas interference among users of the 2.4 GHz band is commonplace [122]. However in an agricultural situation the author believes that the chance of significant interference is low.

3.5 Link budgets and antennas

In any radio system there is a link budget. That is the total amount of loss that can be sustained between transmitter and receiver while keeping packet losses to an acceptable level. In general for a channel without significant interference the amount of loss that can be sustained is determined mainly by the output power of the transmitter and the sensitivity and internal noise of the receiver.

It might naively be assumed that it is possible to determine the acceptable loss by taking the difference between the transmit power and receive sensitivity figures given in the data sheet. For example the MRF24J40 has a nominal transmit power of 0 dBm and a receive sensitivity of -95 dBm, a difference of 95 dB. In practice the loss that can be sustained with acceptable performance is around 10 dB to 20 dB lower than this [43].

The Plane Earth propagation model [84, pp 23] for propagation over a flat reflective surface (such as the earth) where the separation between the antennas is much greater than the height of either antenna is given in non-decibel form by Equation 3.1. P_R is the receive power, P_T is the transmitted power, G_T is the transmitter antenna gain, G_R is the receiver antenna gain, h_T is the height of the transmit antenna, h_R is the height of the receive antenna and d is the separation between the antennas. The two power values must be in the same units and likewise the three length measurements must be in the same units.

$$\frac{P_R}{P_T} = G_T G_R \left(\frac{h_T h_R}{d^2} \right)^2 \tag{3.1}$$

This equation states that for propagation over a flat reflective surface (such as the earth) that the received signal is proportional to the fourth power of antenna height (assuming transmitter and receiver antenna heights are equal) and inversely proportional to the fourth power of the distance between the antennas. The Plane Earth model makes a large number of

assumptions and as mentioned in Section 2.5 more complex models are required to accurately predict channel loss in practice.

Crop foliage can be assumed to be lossy, exactly how lossy will depend on many factors. This would seem to point to tall antenna mounts as being the best solution to minimise path loss. However tall antennas have practical issues. A tall antenna mount will require a much stronger anchor at the ground than a shorter one. Furthermore if the antennas are located above the canopy of the crop then care must be taken to prevent damage arising from contact between the antennas and farm equipment. Further complicating this is the fact that the foliage height changes over a growing season.

A possible compromise is to use tall antennas for the links from base station to repeaters in a hierarchical network and then use low level antennas to communicate with end nodes.

Antenna gain is another issue. Clearly higher gain antennas can improve the link budget. However increasing the gain of an antenna nearly always results in an increase in physical size and the gain available with omnidirectional antennas is limited due to the one dimensional scaling. Directional antennas may be able to be used in some scenarios but they need to be aligned with the direction in which communication will take place. This means they can only be used for communication in a single direction and complicates the deployment process.

Another way to improve the link budget is to use a range extender IC such as the Texas Instruments CC2591 [112]. These ICs combine a power amplifier and a low noise amplifier along with signal switching circuitry to activate the power amplifier when the RF transceiver is in transmit mode and the low noise amplifier when the RF transceiver is in receive mode. The CC2591 provides approximately 20 dB of gain in transmit mode and approximately 10 dB of gain in receive mode. To be usable in a low power wireless sensor network a range extender must also have a low power sleep mode.

Link budget issues may also impact the choice of frequency band if one band is found to have better propagation than others through a given crop.

3.6 Modulation types

Modulation types for relatively low data rates can be broadly divided into two categories, narrowband and spread spectrum. Narrowband modulations are simpler but because they put all the signal energy in a very narrow band they are vulnerable to frequency selective fading and narrowband interference. Therefore spread spectrum modulation schemes are preferable.

Spread spectrum modulation techniques can be broadly divided into two categories, direct sequence spread spectrum (DSSS) and frequency hopping spread spectrum (FHSS). In a DSSS system the energy of each bit is spread over the frequency range using a chipping code (a pseudo-random sequence) which is directly combined with the bits to be transmitted while in FHSS a sequence of bits is transmitted at one frequency then the system hops to another frequency to continue transmission. [103, pp 38] DSSS tends to be more suitable for short communications as there is no need to synchronise frequency hops before starting.

IEEE 802.15.4 (the low level protocol used by Zigbee) operating in the 2.4 GHz band uses a DSSS modulation technique [46]. Four information bits are used to select a 32-bit chip sequence. The 32-bit chip sequence is then modulated onto the carrier using O-QPSK with a half-sine pulse shape. The data rate is 250 kb/s and the chip rate is 2000 kchips/s. 16 channels are available with centre frequencies ranging from 2.405 GHz to 2.8 GHz in steps of 5 MHz.

3.7 Sensors

The sensors themselves are also a consideration. To make a low power wireless sensor network useful it needs to be combined with sensors that are both useful and have low power consumption. Generally to be acceptable for use in a low power wireless sensor network, a sensor must be of a type that can be briefly turned on, a reading taken and then turned off again such as a temperature sensor or a soil resistance probe. In particular, sensors requiring heating are unlikely to be suitable for a low power sensor network.

Chapter 4

Antenna investigation

Ideally when setting up a link between two radio transceivers, a pair of high gain directional antennas would be used. Using high gain directional antennas has a number of advantages. Firstly an increase in antenna gain will increase the signal strength on the direct path and therefore will increase maximum distance that the system can operate reliably over. Secondly an increase in antenna gain may reduce the signal strength of some (but not all) indirect paths. When a signal arrives via more than one path the time-shifted copies of the signal can interfere with each other.

However directional antennas have practical issues that make them an undesirable choice for wireless sensor networks. Firstly they tend to be physically large in at least two dimensions with larger gain usually implying larger size. Secondly they must be aimed during deployment complicating the deployment process. Thirdly they are only useful for point to point links, nodes in a wireless sensor network usually need to communicate with more than one other node. Even for an end node that provides no routing functionality it is desirable for reliability reasons for it to be able to use more than one path to the rest of the network. Finally they tend to be relatively expensive.

At the other extreme monopoles are small, simple to construct and when mounted vertically are omnidirectional in the horizontal plane allowing communication in multiple directions over a flat surface. However they also produce significant radiation outside the horizontal plane. When operated with a large ground plane assembly this radiation is largely directed upwards, however when operated with a very small ground plane assembly significant energy is directed downwards. If the antennas are operating above all foliage then energy directed downwards has a much greater chance of being reflected and interfering with the direct signal than energy directed upwards. On the other hand if the antennas are operating in a gap between foliage and the ground then energy directed both upwards and downwards may be reflected.

The author therefore concluded based on his understanding at the time that if it could be constructed cheaply and robustly the ideal antenna for a wireless sensor network would be a high gain omnidirectional antenna. The author has since realised that reality is more complex than this. For a network where the antennas are located above the crop, where the ground surface was perfectly flat and where the antenna was perfectly vertical, the

high gain omnidirectional antenna would improve the signal level at the receiver. However in a real system the ground is not likely to be perfectly flat. Along with size and cost considerations for the antenna itself this will limit the gain of a practical wireless sensor network antenna. For a network where the antennas are buried within the crop foliage the foliage can be expected to scatter radio signals. The author believes that this is likely to reduce or eliminate the improvements from a higher gain antenna.

One option for constructing an omnidirectional antenna with more gain than a simple monopole or dipole is a collinear array. In such an array multiple radiating elements are lined up vertically and configured so the currents in them are in phase reinforcing the signal in the horizontal direction. To maintain currents in the elements in phase either they must be fed separately or a structure to alter the phase must be inserted periodically along the element, conventionally at half wavelength spacing. The principle was originally described by Franklin who achieved the phase shifts by concentrating alternate half wavelengths of a long wire into a small space by either doubling them back or winding them as inductance coils [38]. Other approaches include using multiple sections of coaxial cable with the core of one section connected to the outer of the next and vice-versa [51] and microstrip lines on a PCB with loops to act as phasing stubs [105].

In the early stages of the project before the author was aware of this theoretical background an antenna design by Martin Pot described as a "Home-brew Compact 6dBi Collinear Antenna"[88] was found on the web. This antenna was made by bending wire and will hereafter be referred to as the "bentwire collinear". The antenna design had three straight sections with two loops in between. The antenna was designed for a centre frequency of 2.45 GHz with the section nearest the connector half a wavelength (61 mm) long, the central section three quarters of a wavelength (91.5 mm) long and the section furthest from the connector 83 mm (just under three quarters of a wavelength) long. It was not stated why three quarters of a wavelength was chosen rather than the half wavelengths described by Franklin. The exact size of the loops was not specified, merely a suggestion that they should be 15 mm or less so that the completed antenna could be enclosed in 20 mm conduit for protection. No discussion of ground planes was made but since the antenna is unipolar like a monopole the grounding arrangements or lack thereof will clearly have an impact on it's behaviour.

It was decided to investigate this antenna and compare it's gain with a simple monopole through both simulations and measurements in an anechoic chamber. This was done to examine the possibility of using them in a wireless sensor network though in the end neither design was used in the field experiments. Ultimately, this investigation did not produce any relevant innovations or discoveries but it is documented here because the experience gained from it influenced the direction of the project.

In order to measure the gain of the wire antennas, an experimental arrangement allowing transmission through the wire antenna was required. For this arrangement another antenna was required for use as the "test antenna". Two sizes of waveguide horn antenna designed for different sizes of waveguide were available in the laboratory, having been constructed for previous projects. Waveguide horns make good test antennas because they do not require a ground plane to operate against and are highly directional making them relatively immune to the effects of clutter that is not directly in front of the antenna. Experiments were first

performed to determine which (if either) of the two sizes of waveguide were suitable for use in the 2.4 GHz ISM band. Then the horns of the chosen size had their gain characterised through both simulation in HFSS and measurement in an anechoic chamber. The results of the simulation and measurement were compared and their close correspondence gave the author confidence in the results.

It was also intended to use the horn antennas for field measurements. To support this when telescopic stands were built for use in field measurements provision was made for supporting the horn antennas on them. Details of these stands are discussed in Section 6.3. However, the size and weight of these antennas made handling them awkward, and, combined with the constraints on how much time was available to work in the rapeseed field, they were not actually used outside the laboratory.

Having characterised the gain of the waveguide horns, the gain of the two different types of wire antennas were obtained through measurement and simulation. However, the differences in gain between the simulation and experimental results were of similar magnitude to the differences between types of antennas. This led the author to abandon this line of investigation. It was later discovered that these differences were partially due to flawed comparisons.

Separately at the end of the project it was decided to measure the return loss of the antennas that had been used in the field work. The return loss indicates what proportion of the energy delivered to the antenna is accepted by the antenna and what proportion is returned to the source. However it does not indicate what direction that energy is radiated in.

4.1 Waveguide horns

A waveguide is a metallic tube used to carry radio frequency signals. [108, pp 18-3] Typically, waveguides are rectangular in cross-section to control the polarisation of the signal. Unlike a coaxial cable which can work from DC all the way to its maximum frequency, waveguides only provide stable, low-loss transmission characteristics within a limited frequency band. If the frequency is too low then the signal cannot propagate along the waveguide at all. If the frequency is too high, then higher order modes of excitation will become able to propagate in the waveguide and the transmission characteristics of the waveguide will become erratic as those modes interfere constructively at some frequencies and destructively at others. The frequency range is determined by the dimensions of the waveguide, with larger waveguides being used for lower frequencies. Sections of waveguide are typically rigid and fitted with flanges that are bolted together to make connections.

To connect equipment with coaxial connections to equipment with waveguide connections, a device known as a waveguide transition is used. In most waveguide transitions, the end of the waveguide is blocked by a metallic end piece. A coaxial connector is then mounted on the side with the outer conductor of the coax joining to the metal of the waveguide and the inner conductor of the coax connected to a pin within the waveguide cavity. The shape of this pin affects the performance of the waveguide transition. So good quality waveguide transitions have the pin geometry carefully controlled.

Two sizes of waveguide horn antenna were available in the laboratory. They were built to similar designs but scaled to match two waveguide sizes, WG10 and WG11A [47]. For both these sizes of waveguide the 2.4 GHz ISM band was outside the accepted frequency range of the waveguide size. However the author was advised that the accepted frequency ranges for waveguide sizes were somewhat conservative and that in practice frequencies just outside the accepted range could sometimes be used.

To determine which waveguide size was more likely to be suitable for the 2.4 GHz band and to characterise the loss in the transitions so that the gain of the antennas alone could be calculated, the waveguide transitions for these sizes of waveguide were measured on an Agilent E5071B 4-port vector network analyser (VNA) with a frequency range of 300 kHz to 8.5 GHz. While performing these tests the opportunity was taken to also measure WG12 and WG14 transitions.

4.1.1 Waveguide transition characterisation

The waveguide transitions were measured with one or more sections of waveguide in between to suppress any higher order modes. To remove the effect of the cables from the measurements, the VNA was calibrated using an Agilent N4431-60003 electronic calibration box. For the WG10 tests, two pieces of rigid waveguide each 25 cm long were used. For the other sizes the details of the sections of waveguide were not recorded. Figure 4.1 shows one of the WG10 transitions connected to one of the lengths of waveguide.

The waveguide sizes tested were WG10, WG11A, WG12 and WG14 (these being the sizes of waveguide readily available in the laboratory). Graphs of the frequency responses (in dB) of two transitions with waveguide in between are shown in Figure 4.2. The graphs are normalised so that the peak value is 0 dB. From each set of measurements an approximate -3 dB bandwidth was determined. The results of these tests are shown in table 4.1 along with information on the names, dimensions and accepted frequency range for each waveguide size taken from Reference Data for Radio Engineers [47, pp 25-8 to 25-9].

From these tests it became clear that while 2.4 GHz was slightly outside the accepted range of WG10 it was nevertheless usable at that frequency. The other sizes of waveguide were clearly unusable at 2.4 GHz.

4.1.2 Measurement and simulation of WG10 horns

Measurements of the gain of the WG10 horns were performed by putting a pair of identical antennas in an anechoic chamber. The chamber is 4 m by 3.4 m by 3 m and is designed for a frequency range of 500 MHz to 50 GHz. At one end of each horn is a flange for attaching to other waveguide hardware. This is followed by a section of WG10 sized rectangular tube. The horn then widens out to an aperture 9.9 in by 5 in over a length of 6.82 in. The horns are made of copper clad FR4 with soldered seams. One of the horns is pictured in Figure 4.3.

The horns were fitted with the same waveguide transitions and lengths of waveguide that were tested previously. The VNA was calibrated using a simple through coupling calibration to remove the losses in the coaxial cable from the measurements. Through coupling

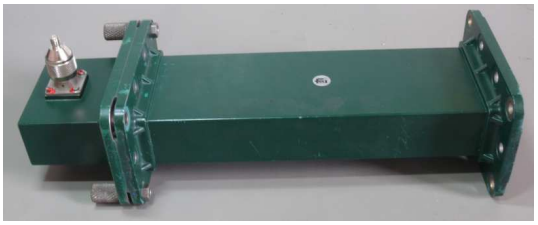


Figure 4.1: A WG10 transition along with a section of waveguide and a N to SMA adaptor

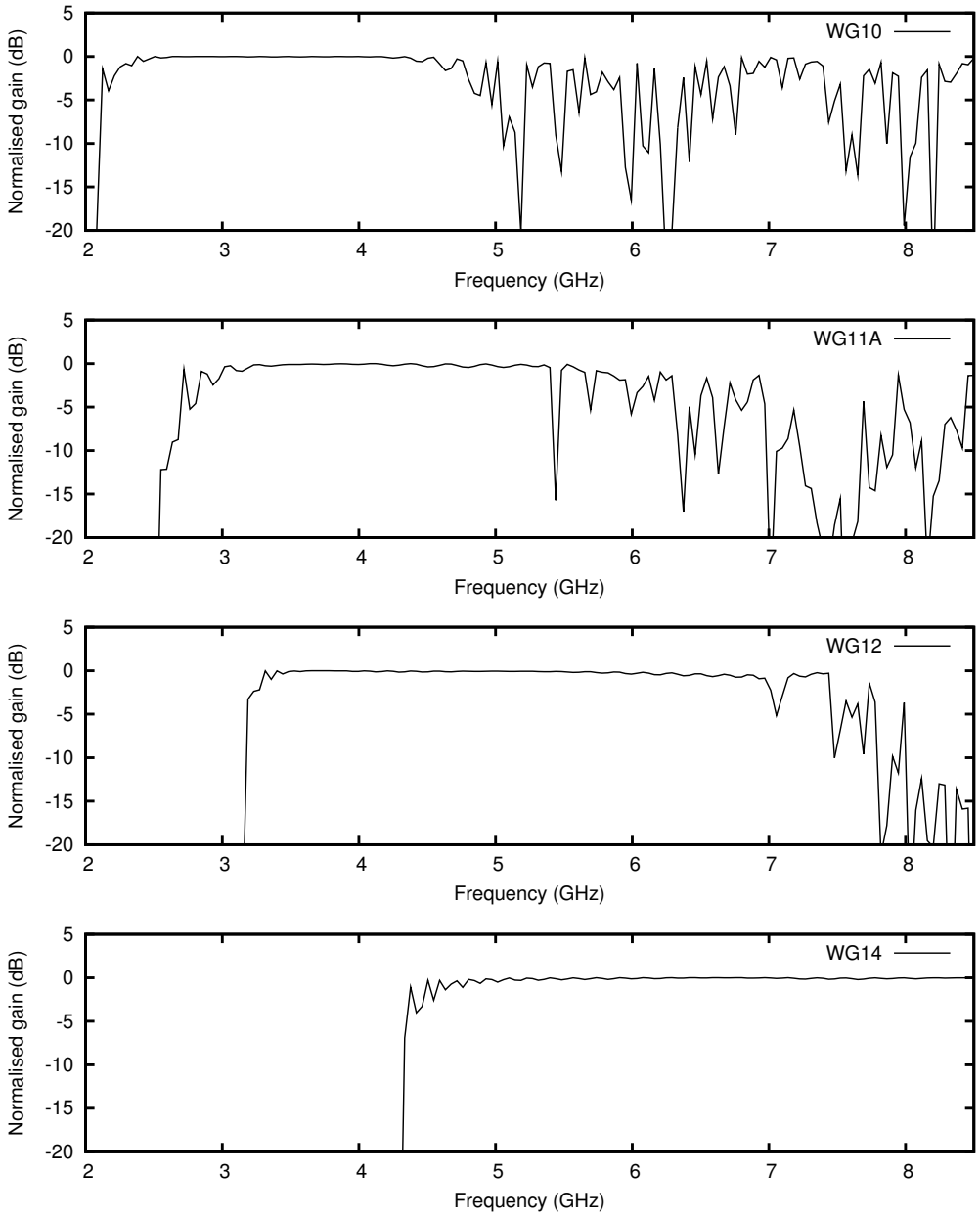


Figure 4.2: Waveguide normalised frequency response

British	IEC	EIA	Outer dimensions and wall thickness	Frequency range from reference	-3 dB Frequency range from tests
WG10	R32	WR-284	3.000 in x 1.500 in 0.080 in thick	2.60 GHz to 3.95 GHz	2.2 GHz to 4.8 GHz
WG11a	R40	WR-229	2.418 in x 1.273 in 0.064 in thick	3.30 GHz to 4.90 GHz	2.85 GHz to 5.4 GHz
WG12	R48	WR-187	2.000 in x 1.000 in 0.064 in thick	3.95 GHz to 5.85 GHz	2.2 GHz to 7.0 GHz
WG14	R70	WR-137	1.500 in x 0.750 in 0.064 in thick	5.85 GHz to 8.20 GHz	4.5 GHz to beyond 8.5 GHz range of VNA

Table 4.1: Waveguide types and parameters

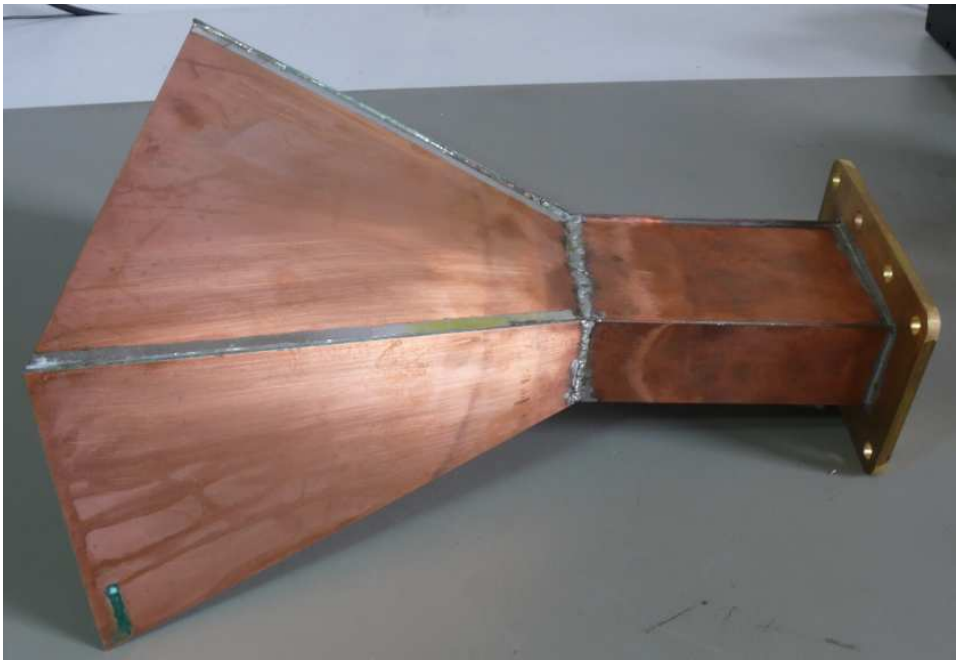


Figure 4.3: The waveguide horn used as a test antenna

calibration was considered adequate because in this application, only magnitude is being measured not phase. The measurements were performed over the frequency range 2.1 GHz to 4.1 GHz, which was chosen because it was roughly the range over which the waveguide was usable. This frequency range differs slightly from the one given in table 4.1 because it was determined by inspecting the graph rather than applying specific criteria.

The Rayleigh distance is $\frac{2D^2}{\lambda}$, where D is the size of the largest feature of the radiating part of the antenna [74] (for horns this can be taken as the longer side of the horn's aperture), and is commonly taken to define the boundary of the far field. The long side of our horn's aperture is approximately 0.25 m and the wavelength ranges from 0.073 m to 0.14 m. This gives a Rayleigh distance varying from 0.89 m to 1.71 m.

At the time, the author assumed that there needed to be a point between the antennas that

was in the far field for both antennas and as such ideally they should be at least 3.42 m apart. However, due to physical constraints in the chamber, this could not be achieved and the antennas were placed only 1.82 m apart. Much later, the author read that a single Rayleigh distance between the antennas is considered sufficient[74] and the configuration used meets that criteria.

The VNA provides the decibel magnitude of S_{21} (this will be referred to as just S_{21} in the equations). This represents the gain in signal level through the system being measured. Since the antennas and the space between them are a passive system, the values for S_{21} are all negative representing a loss in signal level. From these values, the gain of the antennas under test needed to be derived. Since the tests were being performed in an anechoic chamber, free space propagation could be assumed. The free space propagation equation (in decibel form) is shown in Equation 4.1 [39] where P_r is the received power, P_t is the transmit power, G_t is the transmit antenna gain, G_r is the receive antenna gain, d is the distance between antennas and λ is the wavelength.

$$P_r = P_t + G_t + G_r - 20\log\left(\frac{4\pi d}{\lambda}\right) \tag{4.1}$$

To calculate the gain, it is necessary to rearrange the above equation and make some substitutions. S_{21} is equivalent to $P_r - P_t$. Furthermore G_t and G_r are equal since the two antennas are assumed to be the same. Making these substitutions and rearranging to make $G = G_t = G_r$ the subject of the equation results in Equation 4.2.

$$G = \frac{S_{21} + 20\log\left(\frac{4\pi d}{\lambda}\right)}{2} \tag{4.2}$$

The horn was simulated in Ansoft HFSS. To perform the simulation, a 3D model was built of the antenna. Horizontal and vertical symmetry was used in the model to reduce the complexity by a factor of four. The model was then simulated in adaptive mode at 4.1 GHz with the adaptive simulation run for as many passes as the 48 GB of memory in the computer used for simulation would allow. Each pass of the adaptive simulation refined the simulation mesh by adding extra points in areas where HFSS determined the error is greatest. Once the adaptive simulation had refined the model to the maximum extent practical, a frequency sweep was run from 2.1 GHz to 4.1 GHz matching the frequency range used in the measurements.

The simulation ended in waveguide while the physical antennas had a transition attached. As a result of this the loss of a waveguide transition had to be added to the gain calculated from the measurements to get the results into a form that was comparable with the simulation result. Unfortunately, the measurements of the waveguide transitions were performed with a different frequency step to the antenna measurements. To compensate for this discrepancy the measurements were loaded into the simulation software Agilent ADS and exported with a different frequency step causing ADS to interpolate the data.

The gain calculated from the anechoic chamber measurements of the pair of horns and the gain results from the HFSS simulation of the horn can be seen in Figure 4.4. The results were within 0.5 dB of each other from 2.26 GHz upwards. However, at low frequencies the difference became much larger. This is believed to be due to a combination of the poor

behaviour of the waveguide transition in that frequency range and the interpolation that had to be applied to the analysis of the waveguide transitions.

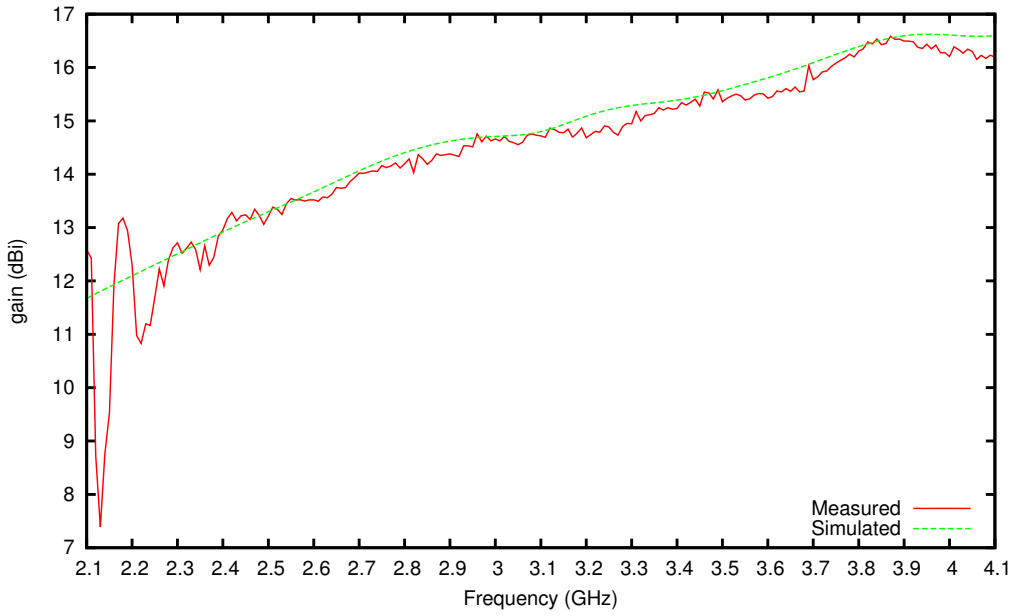


Figure 4.4: Waveguide horns, simulation vs anechoic chamber testing

4.2 Wire antennas

After characterising the waveguide horns, an attempt was made to measure and simulate the gain of two different wire antenna designs both with and without a ground plane disc. One design was a simple three-quarter wave monopole. The other was the "bentwire collinear" design mentioned at the start of this chapter. The antennas were measured and simulated both with and without a ground plane disc which was 400 mm in diameter.

As mentioned previously the "bentwire collinear" design had three straight sections with two loops in between. The antenna was designed for a centre frequency of 2.45 GHz with the section nearest the connector half a wavelength (61 mm) long, the central section three quarters of a wavelength (91.5 mm)long and the section furthest from the connector 83 mm (just under three quarters of a wavelength) long. The antenna used was made of wire approximately 1.5 mm thick with loops having an outside diameter of 13 mm.

Simulation was performed using HFSS which gave a gain figure directly. Measurements were performed in the same anechoic chamber used for the horn measurements. The ground-plane disc for physical tests was made of foil covered cardboard and in the simulations it was represented by an aluminium disc. The two antennas and the ground-plane disc are shown in Figure 4.5. The monopole was three quarters of a wavelength long at 2.45 GHz.

To determine the gain of the antenna, the propagation through a system consisting of the antenna under test and one of the WG10 horns measured previously was measured with the VNA. Then the impact of free space path loss and the gain of the test antenna were

combined with those figures to calculate the gain of the antenna under test. At the centre frequency the antennas were designed for of 2.45 GHz; the monopole without ground plane had a simulated gain of 0.1 dB and a measured gain of -3.6 dB giving a difference of 3.7 dB. The monopole with ground plane disc had a simulated gain of -2.1 dB and a measured gain of 1.3 dB giving a difference of 3.4 dB. The bentwire collinear without ground-plane had a simulated gain of 2.2 dB and a measured gain of 0.6 dB giving a difference of 1.6 dB. The bentwire collinear with ground-plane disc had a simulated gain of 5.5 dB and a measured gain of 2.0 dB giving a difference of 3.5 dB.

The differences discovered between measurement and simulation were of similar magnitude to the differences between the two antenna types. Furthermore as discussed earlier it was believed at the time that the anechoic chamber was too small. These factors left the author with no confidence in either the measured or simulated results. As a result of this it was decided not to make investigation of antennas a focus of the project.

Much later a flaw in the comparison was discovered. HFSS does not include losses due to reflections as part of the "gain" figure it produces while the procedure used for calculating gain from measurements did include such losses. HFSS has a separate option to calculate "realised gain" which produces a figure which includes these losses and is therefore more comparable with the gain figures calculated from the measurements.

At 2.45 GHz the monopole without ground plane disc had a simulated realised gain of -2.4 dB. This was 1.2 dB higher than the measured gain. The monopole with ground plane disc had a simulated realised gain of -1.5 dB. This was 0.6 dB higher than the measured gain. The bentwire collinear without ground plane disc had a simulated realised gain of 1.6 dB, this was 1.0 dB higher than the measured gain. The bentwire collinear with ground plane disc had a simulated realised gain of 4.9 dB. This was 2.9 dB higher than the measured gain.

The results from measuring and simulating the antennas across the range 2.1 GHz to 4.1 GHz are shown in Figures 4.6 to 4.9. Results for the 2.4 GHz ISM band only are shown in Figures 4.10 to 4.13.

The shapes of the measured and simulated realised gain results show some similarity but there are substantial differences. Firstly the measured results show substantial fluctuation with small changes in frequency. This is especially pronounced with the bentwire collinear which shows a variation of approximately 3 dB over the 2.4 GHz band when used with the ground plane disc and a variation of 3.5 dB over the 2.4 GHz band when used without a ground plane disc. Fluctuations with the monopole were smaller at approximately 2 dB without the ground plane disc and approximately 2.5 dB with the ground plane disc.

The author speculates that the fluctuations may be due to differences between the idealised geometry used in the simulation and the actual realities of what was constructed. Differences in the shape of nearby grounded objects may have also played a role, especially in the measurements without a ground plane disc.

The bentwire collinear with ground plane disc showed the largest difference between simulation and measurement in the 2.4 GHz band. Inspecting the graph of the full measurement range showed that the gain peaked at a lower frequency in measurement than in simulation. It is suspected that this was caused by inaccuracies in the construction of the antenna.

Both the measurements and simulations indicate that the bentwire collinear design gave better performance in the 2.4 GHz band than the monopole. However it is doubtful whether the increased size and complexity of construction is justified by the improvements in gain. The results also indicate that including a ground plane disc is likely to improve performance but the practical issues of deploying one make this unlikely in a sensor network application.

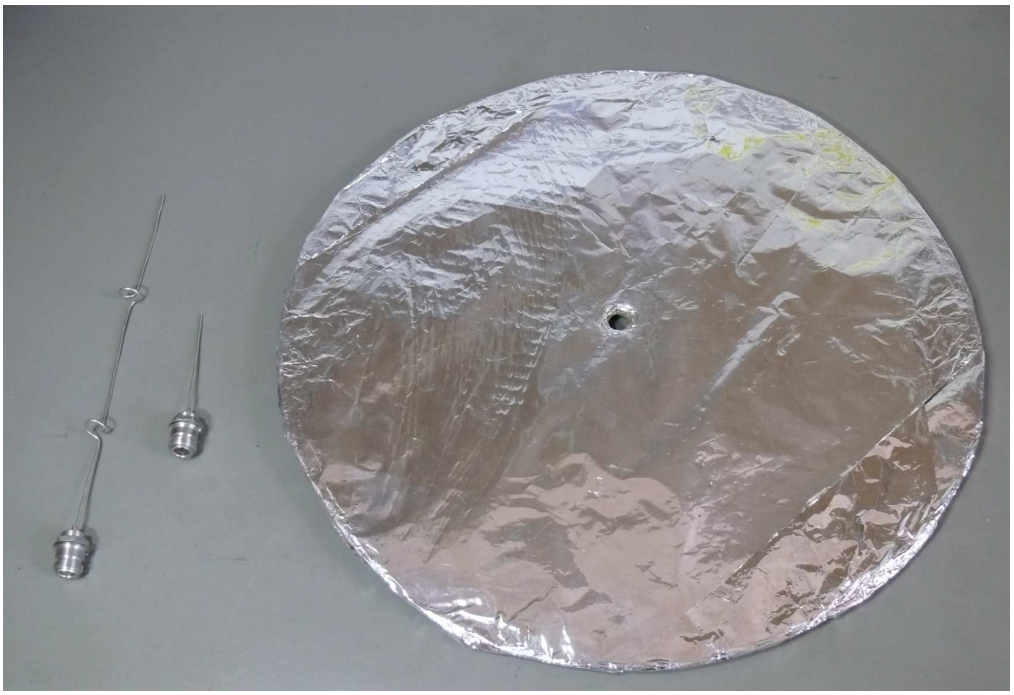


Figure 4.5: The two wire antennas and the associated ground-plane disc

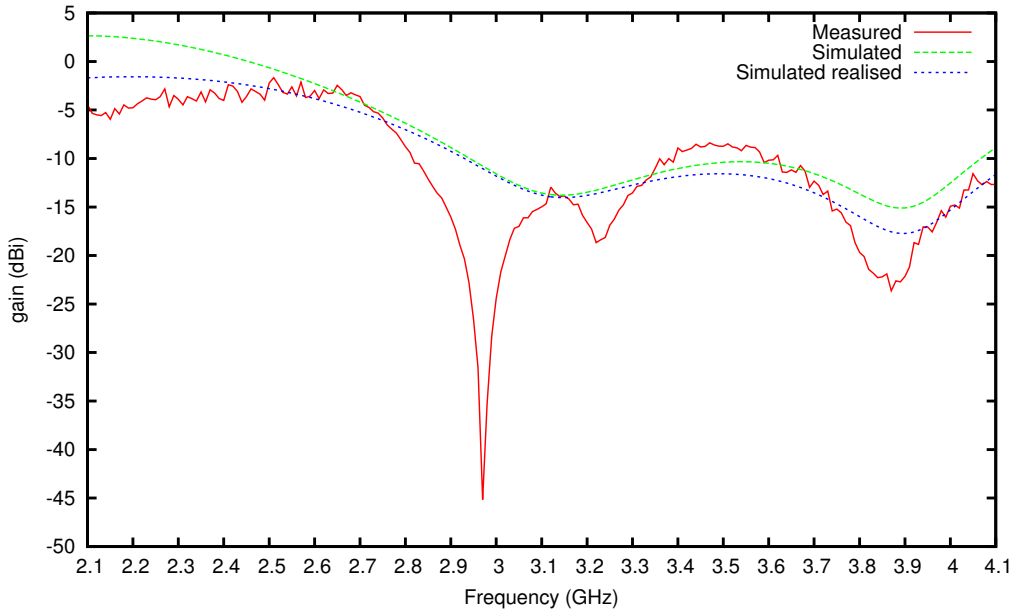


Figure 4.6: Monopole antenna without ground plane disc, simulation vs anechoic chamber testing

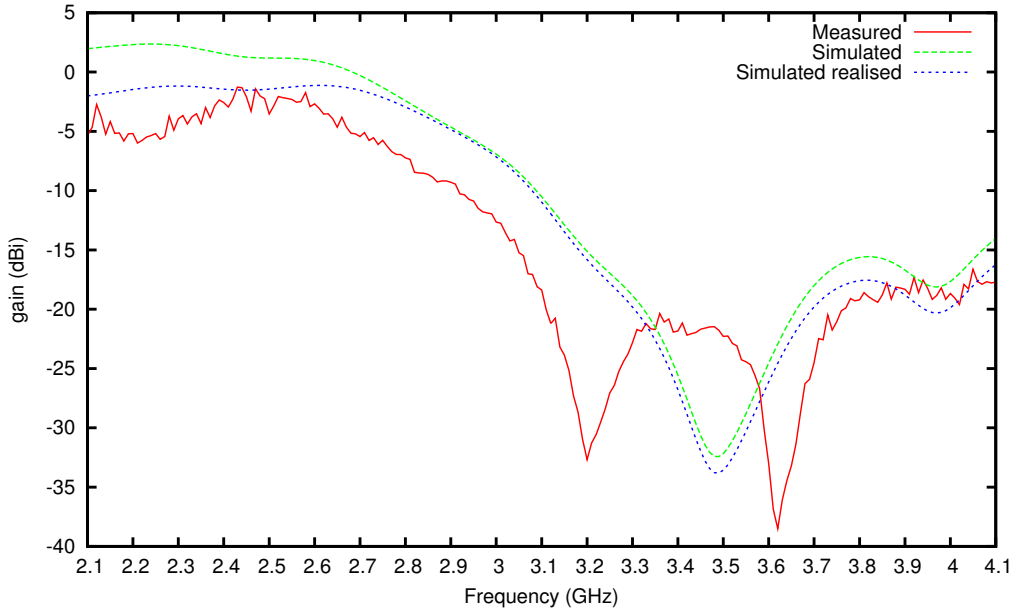


Figure 4.7: Monopole antenna with ground plane disc, simulation vs anechoic chamber testing

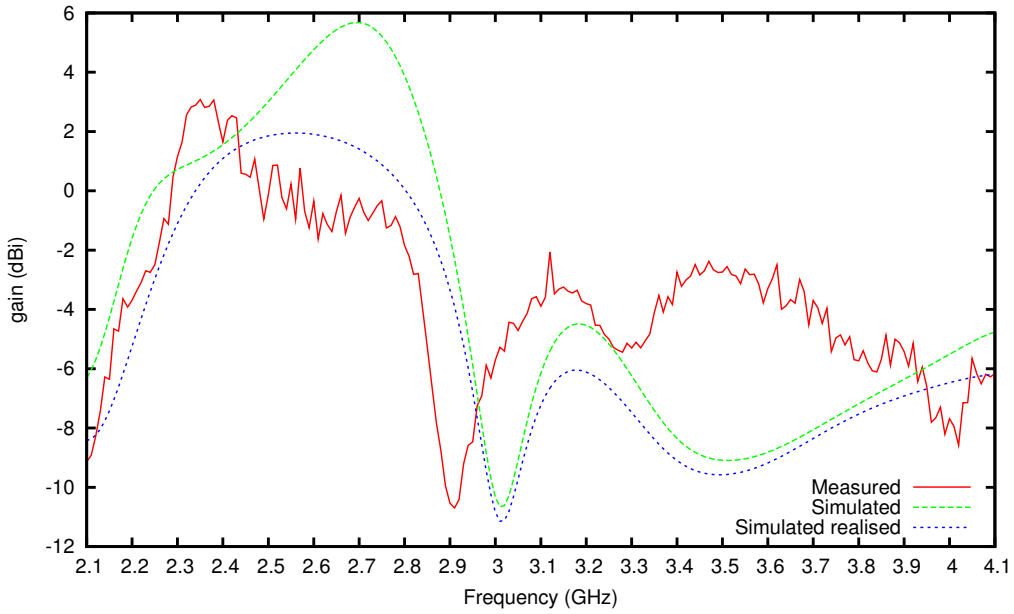


Figure 4.8: Bentwire collinear without ground plane disc, simulation vs anechoic chamber testing

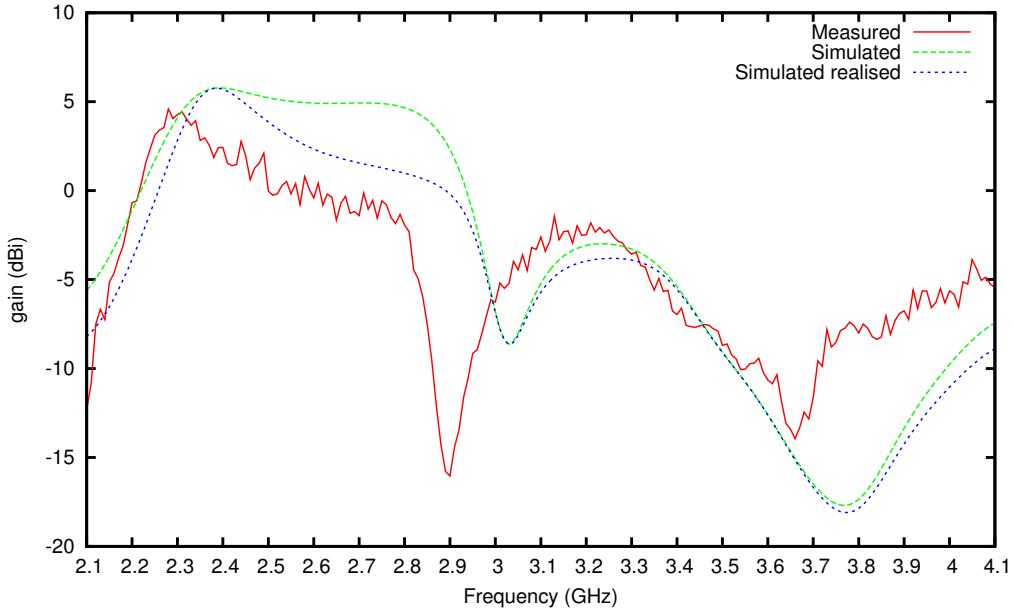


Figure 4.9: Bentwire collinear with ground plane disc, simulation vs anechoic chamber testing

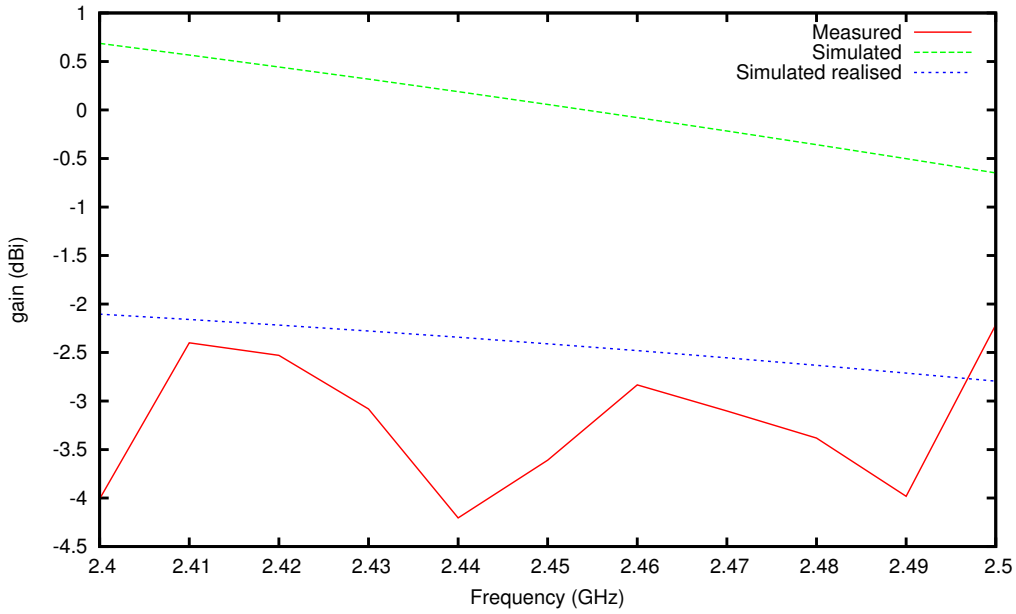


Figure 4.10: Monopole antenna without ground plane disc, simulation vs anechoic chamber testing, 2.4 GHz band only

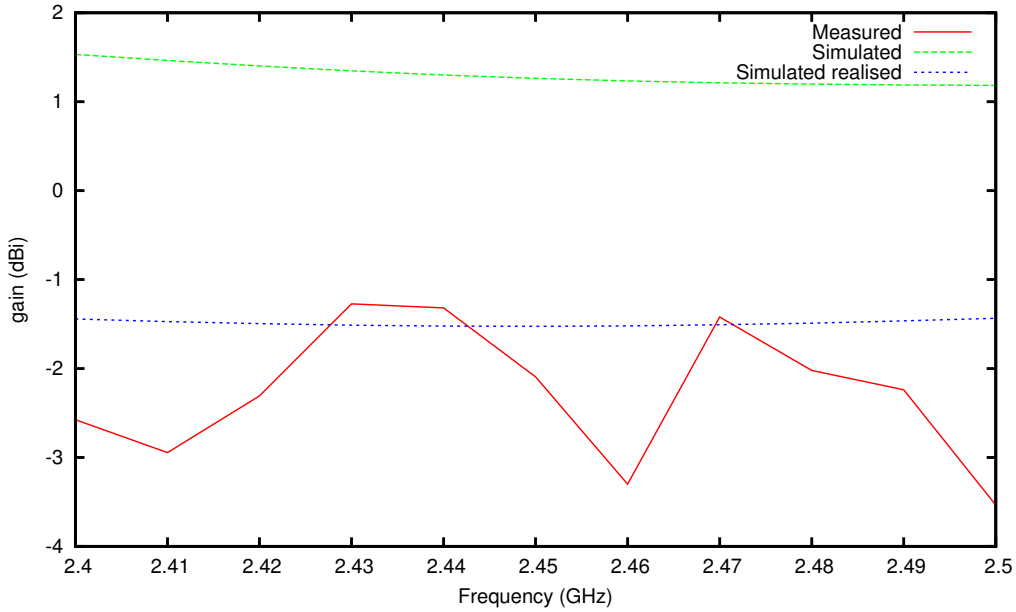


Figure 4.11: Monopole antenna with ground plane disc, simulation vs anechoic chamber testing, 2.4 GHz band only

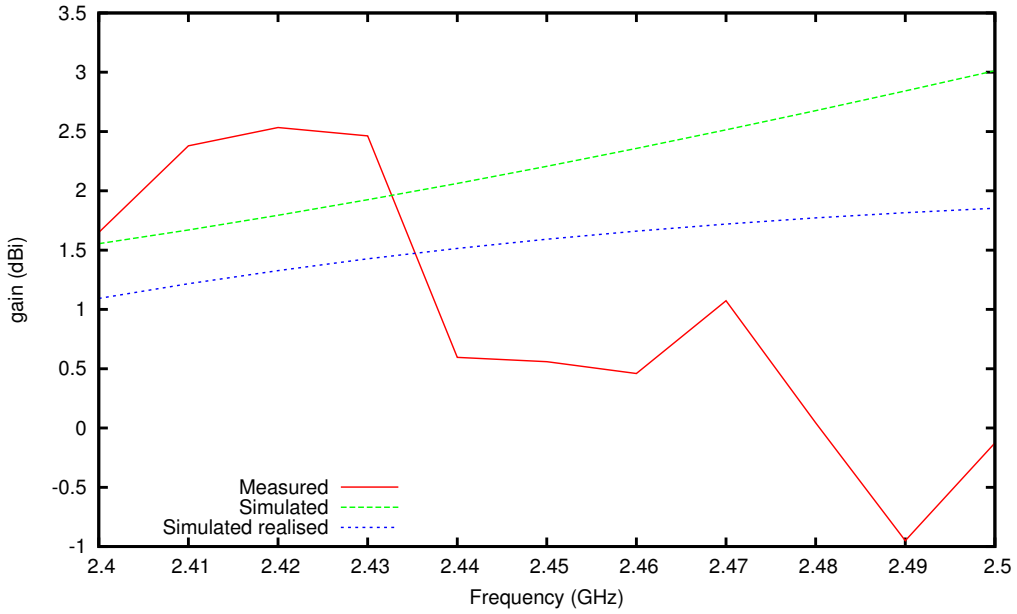


Figure 4.12: Bentwire collinear without ground plane disc, simulation vs anechoic chamber testing, 2.4GHz band only

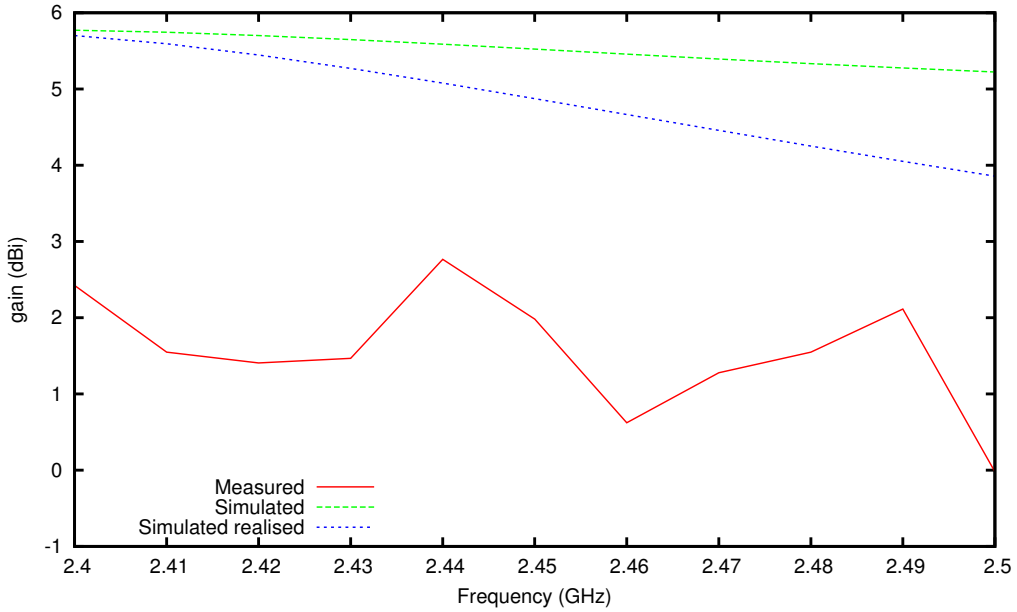


Figure 4.13: Bentwire collinear with ground plane disc, simulation vs anechoic chamber testing, 2.4GHz band only

4.3 Reflection tests on antennas used in field work

At the end of the project the reflection characteristics of all the antennas used in the field work were measured. These measurements determine what proportion of the power from the test equipment is accepted by the antenna and what proportion was reflected back to the signal source. Three antennas were measured. These will be referred to as the "sprung monopole", the "improvised coax monopole", and the "final coax monopole".

The "sprung monopole" was the first to be constructed. The idea of this antenna was it would be able to survive being knocked over. The radiating element of the antenna was formed from a spring, and a wire both soldered to the centre contact of a N connector. A carbon fibre rod sat between the wire and the spring and provided rigidity. The radiating element was sized to be a quarter wavelength at 2.4 GHz. This design was used in some initial experiments during the field trip to Rothamstead Experimental Station in 2011 described in Section 8.1 but they were difficult to construct and during the trip the author came to suspect that they may be having a substantial impact on the measurements. Therefore these antennas were not used in any measurements for which results were recorded. Longer variants of this antenna were also made for other frequency bands and wavelength fractions but these were never used.

The "improvised coax monopole" was the second to be constructed. A pair of these antennas were constructed during the field trip to Rothamstead Research Station in 2011 to replace the sprung monopoles. The antennas were made from 75 Ω coaxial cable intended for satellite TV use as this was the best that could be obtained quickly while away from the University. The coaxial cable was fitted to a N connector and the end of the cable was stripped to make a monopole. Both the foil and the centre conductor were left exposed. The antennas were gradually shortened during the field trip to measure different frequency bands until they were finally reduced to be a quarter wavelength at 2.4 GHz.

The "final coax monopole" was used for the deployment and measurement activities at Tatton Dale Farm as documented in Chapter 9. Many copies of this design were constructed to allow for both measurement and deployment use. The antennas were made from lengths of RG-58 50 Ω coaxial cable fitted with N connectors. Once again a monopole was formed by stripping the end of the coaxial cable. The radiating element was just over a quarter wavelength at 2.4 GHz. This length had been chosen in an attempt to minimise reflection during testing with a VNA [1]. The cable length was chosen to place the base of the monopole 150 mm above the top of the node enclosure. The outer insulation was left on the cable to the point where the braid ended and the inner insulation was left on to the end of the monopole. Thick heat shrink was used to keep the construction reasonably close to vertical while thinner heat shrink was used to seal the place at which the outer insulation and braid ended. The very end of the antenna was also sealed with a blob of silicone sealant but this often fell off in practice.

The antennas were mounted on the surface stands described in Section 6.3. These stands were used for the measurements performed near ground level in the field. The vector network analyser was calibrated at the connector on the antenna stand to eliminate the effects of

[1] However those tests were done with the antenna mounted directly on the VNA, not on an antenna stand, so the grounding situation was different from in the tests discussed in this section

the cable to the stand and the right angle adaptor attached to the stand. Measurements were taken over the frequency range 2.4 GHz to 2.5 GHz with 551 points in the sweep. A photograph of the three antennas is shown in Figure 4.14.

Figure 4.14: Antennas used during field work. Left: "Sprung monopole", Centre: "improvised coax monopole", Right: "final coax monopole".

To determine what if any impact the conductivity of the surface the stands were placed on had on the reflection characteristics of the antennas, the tests were done twice, once with the antennas stands placed on an empty cardboard box to keep them away from anything conductive and once with the stands placed on the foil covered cardboard disc that had been used as a ground plane for the tests described in Section 4.2.

The results with the stand placed on the cardboard box were very close between the three

antennas with a difference of just over 1 dB between the highest and lowest readings. Overall the return loss ranged from just under 3.2 dB to just over 4.2 dB. Surprisingly the "final" antenna had about 0.4 dB lower return loss than the other two. These measurements are plotted in Figure 4.15.

Placing the stand on the foil covered disc had a surprisingly inconsistent effect. The return loss for the "sprung" antenna increased by about 1.4 dB, the return loss of the "improvised" antenna reduced by approximately 0.3 dB and the return loss of the "final" antenna reduced by approximately 0.7 dB. Especially surprising was that the antenna with it's radiating element furthest from the stand showed the largest reduction in return loss when the stand was placed on the foil covered disc. These measurements are plotted in Figure 4.16

Overall the return losses in these tests range from 2.7 dB to 5.6 dB. This means that between 27 % and 54 % of the power going into these antennas is reflected back into the source when connected to a 50 Ω transmitter. A similar loss will apply when using the antennas on a 50 Ω receiver. It is likely that this could be improved with matching components at the bottom of the radiating element. However the sensitivity to the environment is likely to limit how well the impedance of wireless sensor network antennas can be matched in practice. Furthermore including matching elements at the base of the radiating element would mean that antennas could not be constructed simply by stripping the end of a coaxial cable.

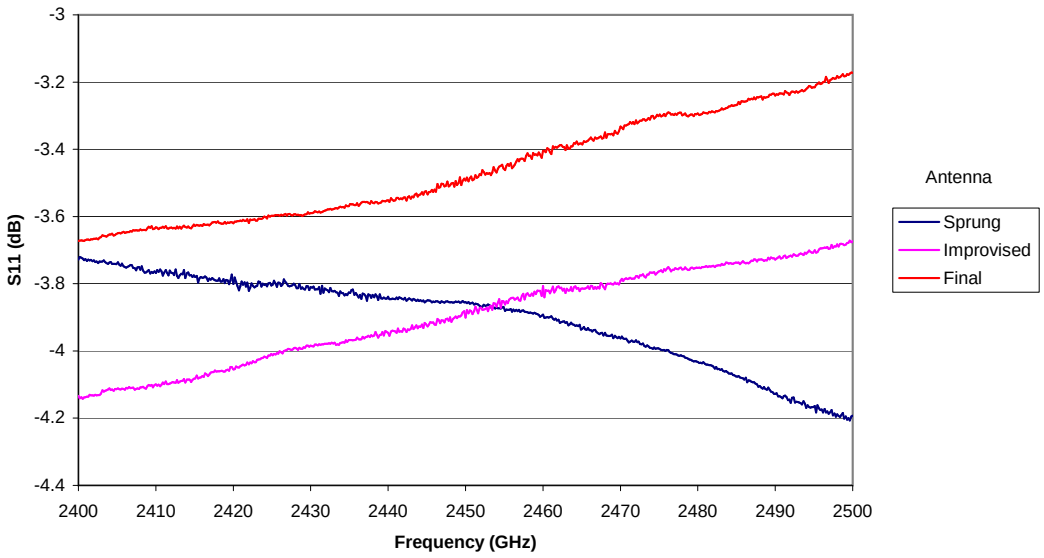


Figure 4.15: Results of reflection tests on antennas used in field work. Antenna stand placed on cardboard box.

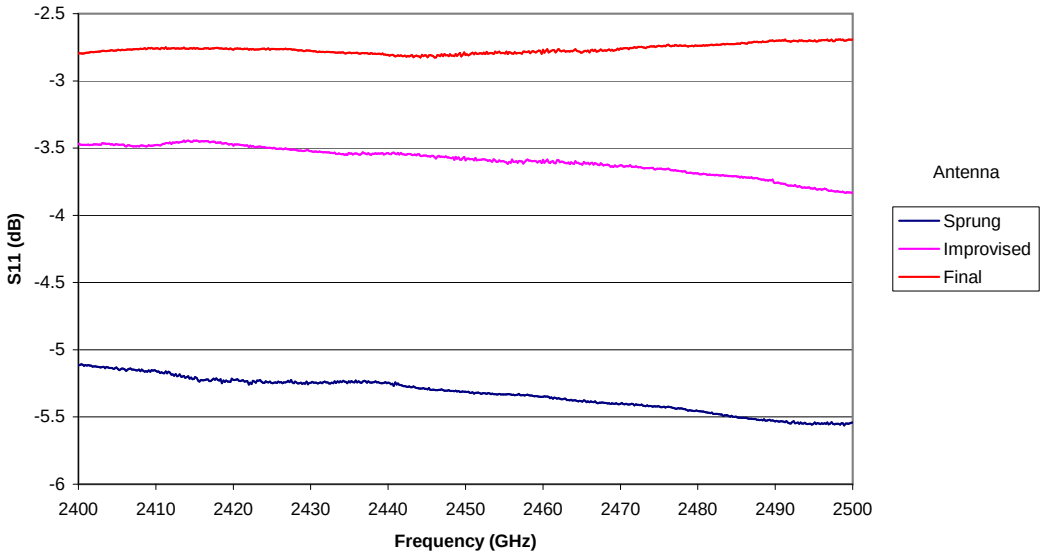


Figure 4.16: Results of reflection tests on antennas used in field work. Antenna stand placed on foil disc.

Chapter 5

Wireless sensor node power measurement system

It is useful to know the power consumption profile of a wireless sensor node. Knowing the peak power consumption is important for the sizing of components in the power supply path while knowing the average power is important for battery sizing. Both peak and average power can be obtained from a power profile. Furthermore, a profile of the node's power consumption combined with knowledge of the software may give insight into where power can be best saved. Given a fixed voltage, power is proportional to current. Therefore, in a test environment with a known supply voltage, a power consumption profile and a current consumption profile provide the same information.

As mentioned in the literature review in Chapter 2, a simple way to record a power profile is to put a resistor in series with the load and measure the changing voltage with a differential probe on a digital storage oscilloscope. However, this technique runs into the problem of dynamic range. When the radio is active, a sensor node can draw tens of milliamps but standby current for some parts can in theory be less than a microamp, though the author later discovered that tens of microamps was more typical for a complete node design. A power measurement system aiming to measure values from 0.5 µA to 50 mA while maintaining 10 % accuracy would need a dynamic range of 10^6 which is far higher than that of a typical digital storage oscilloscope. Searches by the author at the time did not reveal any designs for power consumption measurement with a dynamic range this large.

It would clearly be possible to improve the dynamic range of such a system by replacing the differential probe on the oscilloscope with a precision differential instrumentation amplifier and data acquisition device. This would give better dynamic range than an oscilloscope but it was felt that by itself this was still unlikely to give sufficient dynamic range.

Therefore, a new system specifically for sensor node power measurements was designed and constructed. The key challenge with this system was achieving sufficiently high dynamic range to measure both the standby and operating currents. Accuracy was not considered critical, a measurement within ten percent of the actual value would be more than adequate for sizing batteries and regulators. High frequency response was also unnecessary as the

interesting parts of a nodes power profile (with the possible exception of nodes that use range extenders to increase the transmit power) are typically of the order of milliseconds or more. Furthermore, high frequency currents can be delivered by the bypass capacitors on the device's power rails and do not need to be delivered directly by the batteries or regulators. For example, at 1 kHz a 10 µF capacitor has an impedance of 15 Ω, at 10 kHz; this drops to 1.5 Ω. As such it was decided to aim for a break frequency of 10 kHz but with the understanding that lower break frequencies down to around 1 kHz would be acceptable. Clearly, since the aim of the system is to measure power consumption over potentially long periods of time, the system must be DC coupled with no lower limit on frequency.

A block diagram of the hardware in the system is shown in Figure 5.1 and a diagram of the signal flow through the system is shown in Figure 5.2. To achieve high dynamic range three techniques were used. Firstly, the sense resistor was included in the feedback loop of an op-amp; this allows a large sense resistor (giving a higher output voltage and thus reducing the impact of noise in later stages) to be used without having an unacceptable impact on voltage supplied to the node. Secondly, a data acquisition device was used which gives far more resolution than an oscilloscope. Finally, a concept was borrowed from high dynamic range imaging [29]. Rather than trying to make a single reading covering the entire range, multiple readings were taken with different dynamic ranges and recombined in software. This technique is realised by running three instrumentation amplifiers in parallel with different levels of gain. The readings from the three channels are then digitised, filtered, scaled and recombined to produce a single set of output readings.

A photograph of the main power measurement board is shown in Figure 5.3. A photograph of the complete system including the noise control measures is shown in Figure 5.4.

As mentioned in the literature review in Chapter 2, it was later discovered that Agilent Technologies were making a similar power measurement system in the form of the N6781A source/measurement unit for their N6700 series modular power system [2] but by that time the author's own system was already largely complete. The software driving the Agilent

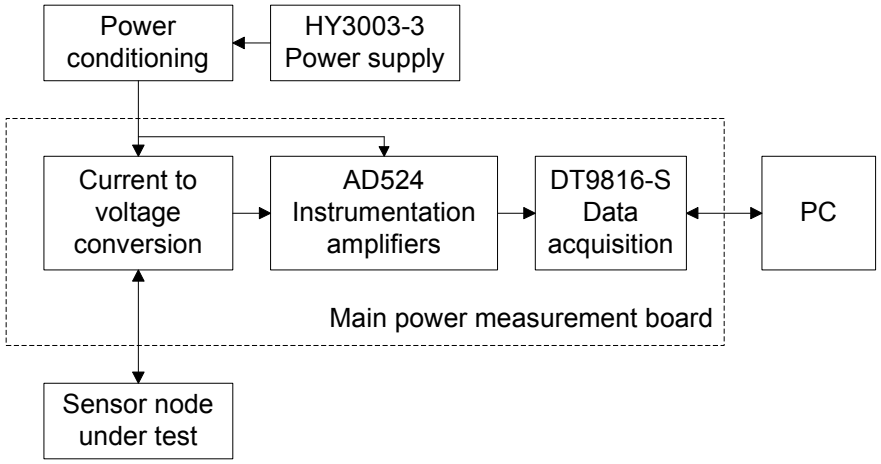


Figure 5.1: Block diagram of power measurement system

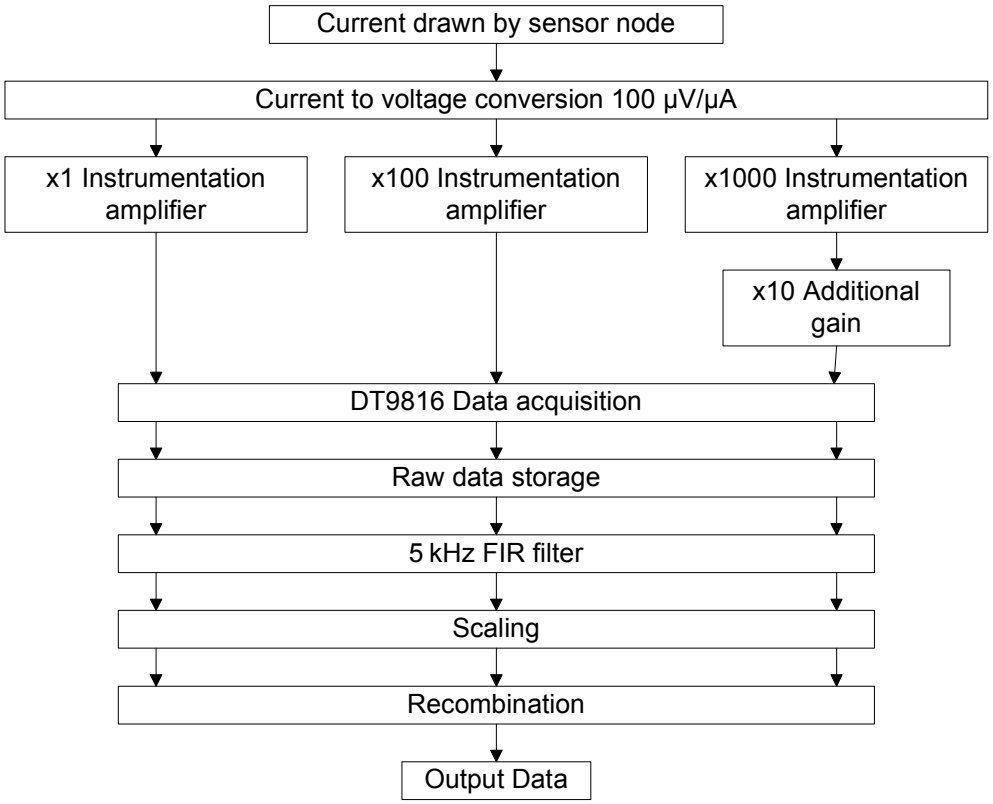


Figure 5.2: Signal flow diagram of power measurement system

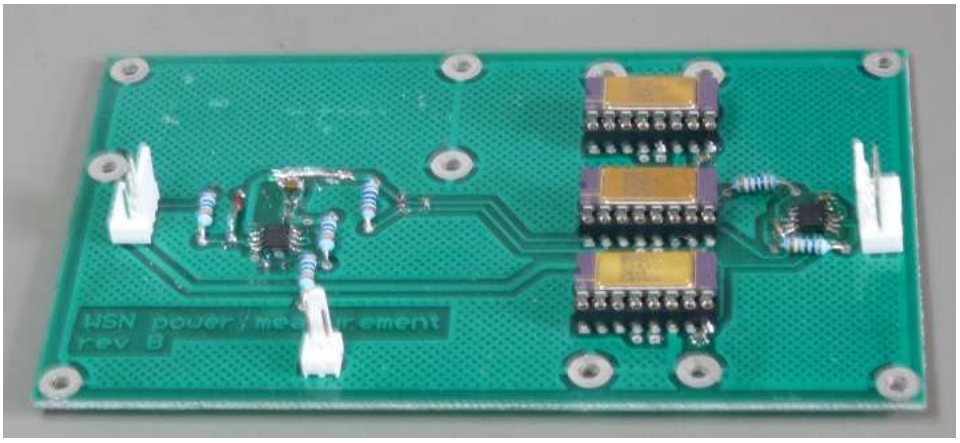


Figure 5.3: Board for wireless sensor node power measurement; on the left is the current to voltage conversion; on the right are the instrumentation amplifiers and the additional x10 gain

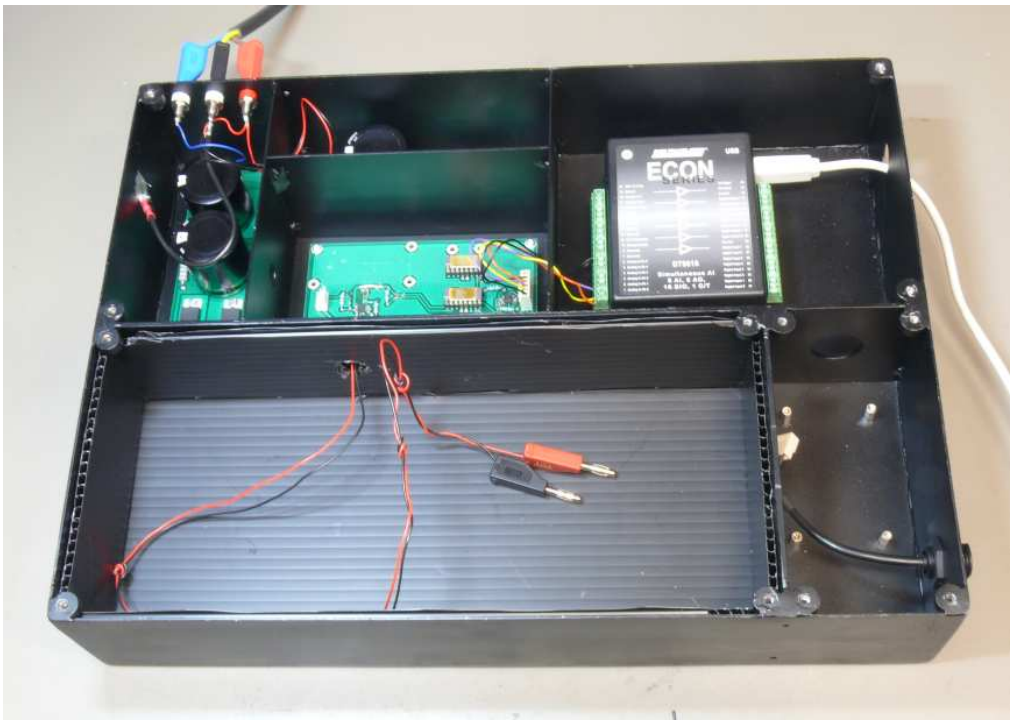


Figure 5.4: Complete power measurement system

system is clearly more refined than that of the system described here and the system also claims to be able to read smaller currents. However, the data sheet for the Agilent system does not appear to mention frequency response.

5.1 Current to voltage conversion

The first section of the power measurement device supplies current to the sensor node under test at a known voltage while measuring the current drawn. A schematic diagram for the current to voltage conversion stage is given in Figure 5.5. The node is powered via an operational amplifier and the sense resistor (R1) is included in the feedback loop of the amplifier. This circuit arrangement means that the voltage across the sense resistor does not significantly affect the voltage supplied to the sensor node and hence allows use of a much larger sense resistor than could be used if simply putting a resistor in series with the node's supply. The sense resistor is $100\,\Omega$ giving a sensitivity of $100\,\mathrm{V/A}$. A capacitor C11 is placed across the sense resistor to form an anti-aliasing filter. This will be discussed in more detail in the next section.

To set the power supply voltage for the sensor node, a reference voltage is fed in from a resistive divider. The divider is built into the cable that connects the main power measurement board to the power filter and regulator board and the voltage is changed by swapping this cable (cables have been made for voltages of 3V, 3.3V and 5V). For early tests, a separate output on the power supply was used to supply the reference voltage. The resistive dividers were introduced to eliminate a possible avenue for noise to enter the system.

It was decided to support currents up to 100 mA into a 3 V node. The author believed this would be sufficiently high to measure the peak power consumption of sensor node hardware without the system saturating. This along with the value of the sense resistor meant that the maximum output of the operational amplifier had to be at least 13 V. Accordingly, it was decided to use a 15 V supply allowing 2 V of headroom for the amplifier.

An AD8397 amplifier [7] was used. This amplifier was chosen because it offered both a high output current (170 mA) and low input bias current (200 nA typical, 900 nA maximum at room temperature). The high output current was needed because the amplifier supplies the power to the sensor node under test. The low input bias current was needed because the input bias current is effectively added to the current being measured. While it is possible to calibrate out some fixed error, a large fixed error could drive the low gain channel of the system into saturation even when there was no load.

C5 helps to block any high frequency noise from the device under test and lowers the output impedance at higher frequencies. The value of 22 µF was chosen to be similar in value to decoupling capacitors typically seen on the power rails of microcontroller circuits.

C12 reduces noise on the reference line; this is important as due to the high capacitance of the output system (both from C5 and from the decoupling capacitors in the device under test) any high frequency noise in the reference signal would be massively amplified and could end up swamping the measurement system.

R7 ensures the system is stable even under very heavily capacitive loads (both from C5 and from the decoupling capacitors in the device under test). It also sets the low frequency impedance seen by the device under test. The value of 1 Ω was chosen to be similar to the internal resistance of batteries.

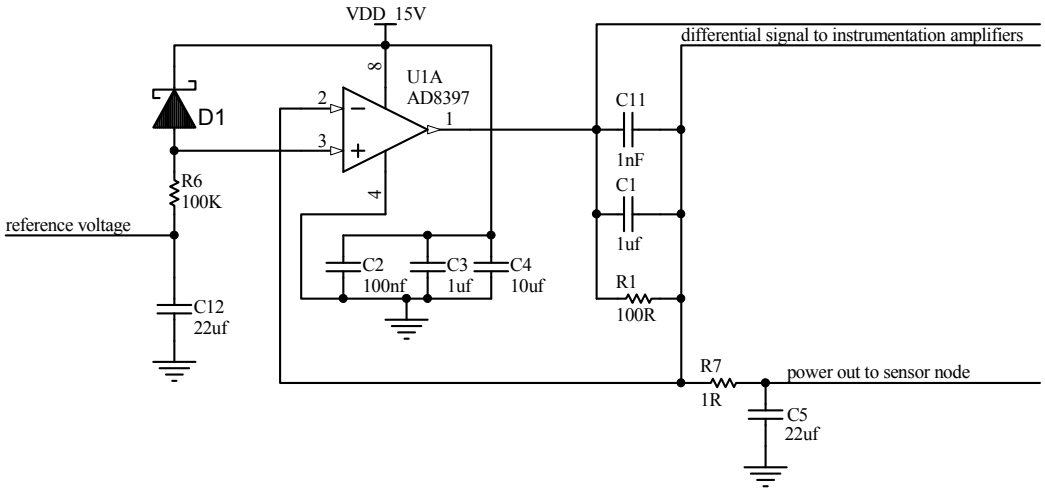


Figure 5.5: Schematic of current to voltage conversion circuit

5.2 Anti-aliasing

The sampling theorem (also known as the Shannon-Nyquist sampling theorem) states that if a band-limited signal is sampled and the highest frequency in the band limited signal is less than half the sampling frequency (f_s) then the samples are a complete representation of the original signal [101]. This limit of half the sampling frequency is referred to as the Nyquist frequency (f_n). If the input signal contains frequency components above the Nyquist frequency they will be aliased to frequencies below the Nyquist frequency. In particular, a signal of frequency f where $f_n \leqslant f \leqslant f_s$ will be aliased by the sampling process to a frequency of $f_s - f$ [40, pp. 301].

Aliasing is harmful to the signal in the frequency range of interest if it converts signals outside the frequency range of interest into signals inside the frequency range of interest. If the highest frequency of interest is f_i then a signal will not cause harmful aliasing if its frequency is less than $f_s - f_i$. To reduce the harmful effects of aliasing, it is normal practice to include an anti-aliasing filter before sampling. Since sampling is a pre-requisite for digitising, this must clearly be an analogue filter.

When designing an anti-aliasing filter, the aim is to minimise the impact on signals within the band of interest, while strongly attenuating signals that will cause harmful aliasing. There is a compromise between the complexity of filter design and construction and the sample rate. The closer the highest frequency of interest is to the lowest frequency that will cause harmful aliasing, the more complex the filter must be to achieve a given level of attenuation of signals that will cause harmful aliasing.

To keep the analogue design as simple as possible, it was decided to use a sampling rate much higher than the desired break frequency of the system. A sampling rate of 100 kHz was chosen which, combined with a desired bandwidth of 10 kHz makes 90 kHz the lowest frequency which will cause harmful aliasing. This in turn meant a simple RC filter was considered sufficient to reduce aliased signals to acceptable levels. A RC filter with a break frequency of 10 kHz will reduce the signal voltage by a factor of approximately nine at 90 kHz, which was considered sufficient.

The filter was realised by placing a capacitor (C1) over the sense resistor (R1). The value of this capacitor should have been calculated using the formula given in Equation 5.1 where f_c is the cutoff frequency in hertz, R is the resistor value in ohms and C is the capacitor value in farads . Unfortunately, a mistake was made and the factor of 2π was not included in the calculation. This led to the calculated value being 1 µF instead of 0.16 µF. This error was not noticed until long after system development was completed and there are currently no plans to modify the system to fix the flaw. A 1 nF capacitor was also added in parallel with the 1 µF capacitor as large capacitors often have poor high frequency behaviour.

$$f_c = \frac{1}{2\pi RC} \tag{5.1}$$

5.3 Instrumentation amplifiers

The differential signal from the sense resistor is amplified and converted to single ended by AD524 instrumentation amplifiers [8]. The AD524 amplifiers were chosen for the combination of low input offset voltage and low input bias current. As discussed at the start of the chapter, three separate amplifiers with different gains are used to measure currents in different ranges and therefore provide a high overall dynamic range. One amplifier is configured with a gain of 1, another with a gain of 100. The final amplifier is configured with a gain of 1000 and followed by a TL071 [106] based amplifier with a gain of 10 to give a total gain of 10000. The TL071 was not considered to be a critical component; virtually any operational amplifier could have been used.

The AD524 comes in a selection of performance grades denoted by a single letter in the part number between the base part number and the packaging option. There are three grades (A-C) for regular temperature applications and one grade (S) for extended temperature range applications. Of the grades intended for regular temperature use, A is the lowest grade and C is the highest grade.

The maximum input bias current of the AD594 varies from 15 nA for the C grade to 50 nA for the A grade. This current was considered negligible in the system design and so input bias current was not a factor in grade selection. The grades also vary in input offset voltage from 50 µV for the C grade to 250 µV for the A grade. This was considered negligible for the medium and low gain channels but significant for the high gain channel, where after amplification by a factor of ten thousand, an input offset of 250 µV would result in an output offset of 2.5 V. Therefore, a C grade amplifier was used for the high gain channel and A grade amplifiers were used for the medium and low gain channels.

The signals are then passed to the data acquisition unit for digitisation. The amplifiers are powered from a ±15 V supply to allow them to output negative voltages. It is important not to have a cut-off at zero volts to prevent noisy or offset signals from being clipped and thereby introducing a bias when the signals are averaged.

5.4 Frequency response testing

To confirm that the frequency response of the current to voltage conversion and associated anti-aliasing filter was acceptable before the data acquisition hardware and software were in place, a signal generator and oscilloscope were used. Two channels on the oscilloscope were connected, one to the output of the signal generator and one to the output of the x1 instrumentation amplifier.

A pair of 100 Ω 1% tolerance resistors connected in parallel to act as a 50 Ω resistor was connected between the signal generator and the measurement system. These resistors serve to translate the voltage output of the signal generator into a varying current presented to the measurement system.

The DC offset on the signal generator was set to match the output voltage of the board as closely as possible to keep DC current to a minimum. Signal levels from the generator were

set to approximately 2 V peak to peak using the scope, after connecting to the measurement system. It was important to set the output signal level after connecting the signal generator to the measurement system due to the signal generator's 600 Ω output impedance. Peak to peak values for both input and output were recorded and from these an approximate gain can be calculated at each frequency. Initially readings were taken at 1 Hz, 10 Hz, 100 Hz, 1 kHz and 10 kHz. These readings determined that the 1 kHz to 10 kHz decade was the most interesting so readings were taken in 1 kHz steps across that decade. Results are given in Table 5.1.

A DC gain of two is expected since the 50 Ω resistor connected to the signal generator gives 0.02 A/V and the 100 Ω sense resistor gives 100 V/A. The actual measured gain at low frequencies was slightly higher than two. It is suspected that this was due to measurement inaccuracies as these measurements were taken with an oscilloscope which is not a precision measurement device.

The break frequency is the frequency at which the square of signal voltage has dropped by a factor of $\sqrt{2}$ to approximately 0.71 times its low frequency value or 3.1 V. We see this happens at between 1 kHz and 2 kHz. This was lower than the break frequency that was aimed for but higher than the lowest break frequency that was considered acceptable. As such, the results were considered acceptable. It was assumed at the time that the lower break frequency was caused by some component other than the anti-aliasing filter limiting frequency but as mentioned in the previous section, it was later discovered that it was due to a design error in the anti-aliasing filter.

Freq	In	Out
1 Hz	2.1 V	4.4 V
10 Hz	2.1 V	4.4 V
100 Hz	2.1 V	4.4 V
1 kHz	2.1 V	3.8 V
2 kHz	2.1 V	2.8 V
3 kHz	2.1 V	2.2 V
4 kHz	2.1 V	1.9 V
5 kHz	2.1 V	1.7 V
6 kHz	2.1 V	1.6 V
7 kHz	2.1 V	1.4 V
8 kHz	2.1 V	1.3 V
9 kHz	2.1 V	1.3 V
10 kHz	2.1 V	1.3 V

Table 5.1: Input and output voltage

5.5 Data acquisition

To digitise the signals from the instrumentation amplifiers and bring them into a computer for digital processing, a data acquisition unit was needed. As discussed earlier, a sample rate of 100 kHz had been chosen to place the Nyquist frequency well-beyond the break frequency

of the system and hence avoid the need for aggressive anti-aliasing filters on the analogue side. Given that the lowest gain channel had a sensitivity of approximately 100 V/A and that it was desired to be able to measure currents up to 100 mA, the data acquisition unit needed to be able to measure voltages up to 10 V.

Given the design goal of 10 % accuracy, it was determined that four significant bits should be available for all readings. For the medium and low gain channels, it was deemed necessary to maintain that precision down to 0.5 % of full scale in order to allow some overlap between the usable ranges for different channels. For the high gain channel, it was deemed desirable to maintain that precision down to an input current of 0.5 µA per original design goals. In both cases, this led to a requirement for 12-bit precision on positive readings.

Noise and offsets in the analogue processing chain could cause negative voltages to be fed to the data acquisition unit. If all negative readings were clamped to zero, then the use of subtraction to remove offsets and averaging to measure values below the noise floor would not operate correctly. Therefore, a bipolar data acquisition unit that can correctly measure these values was required. Recording negative values requires another bit taking the overall resolution requirement to 13 bits.

No suitable data acquisition unit was available for reuse and so one had to be purchased specifically for the system. The data translation DT9816 [28] series was chosen as it met the sample rate and resolution requirements at a relatively low cost. It also offered simultaneous sampling of the input channels, simplifying the recombination process. There are several models in the DT9816 series with different maximum sample rates. Any model in the series would have been sufficient for this system but given the small price difference and the possibility that the device would be re-used in other systems in future, it was decided to purchase the top model of the series, the DT9816-S. The DT9816-S is pictured in Figure 5.6.

Data Translation provide a number of example programs to operate their data acquisition units in different languages. One of these examples, specifically the "Open Layers C example", was used as a base on which to build the program used to acquire data from the power measurement system. The software was modified to change the default settings, to add header information to the output data and to introduce a dialog box for making calibration measurements which are then stored in the header. The software gathers samples in one second blocks and writes them out immediately to disk. The size of the data file is limited to approximately 2 GB. At six bytes per sample point and 100 000 samples per second, this limits the acquisition time to just under 2 hours. This limitation could be removed by rewriting the file output code to use 64-bit file access APIs.

Figure 5.6: Data translation DT9816 data acquisition unit

5.6 Dealing with noise

When the data acquisition unit was first connected to the board with the current to voltage conversion and instrumentation amplifiers, it was discovered that there was a major problem with noise. In particular, the noise on the medium gain channel was so high that there were some input values that would be below the noise floor for the medium gain channel while also causing the high gain channel to be in saturation. To determine the frequency of the noise, Fast Fourier Transforms were run on sample data and it was determined that a substantial proportion of the noise was at 50 Hz and small multiples thereof. The source of the noise was initially unknown with the suspected sources being power supply noise, electromagnetic noise in the environment and noise from the connected PC. Methods to mitigate these three possible sources were worked on simultaneously.

To reduce noise in the power supply, a power conditioning board was designed and built. Firstly each rail was filtered by a RC filter to provide at least 10x attenuation at 50 Hz. The filters were followed by regulators to provide a stable 15V output and hopefully, further reduce noise. Since it was desired to keep the volt drop of the power conditioning board to a minimum, the resistor used had to be kept small. The exact power consumption of the system was not known at this time so a conservative estimate of 1 A was made. A 2.2 Ω resistor was chosen providing a DC drop of 2.2 V at 1 A. The regulators chosen were the National Semiconductor LM2940CT-15 and LM2990T-15 which have a maximum dropout voltage of 0.8 V at a junction temperature of 25 °C. With a ±18 V supply, this was believed to leave sufficient headroom for the regulators to produce a ±15 V output at 1 A current draw. It was later realised that this analysis was flawed since under 1 A load the junction temperature of the regulators would be far greater than 25 °C. However, since 1 A was a significant overestimate of system current draw, this did not pose any problems in practical operation.

The system had originally been powered from a Farnell TOPS 3D power supply, however, this only supplied up to 17V which was insufficient for supplying the power conditioning

board. Therefore, when the power conditioning board was introduced the DC power supply was replaced with a HY3003-3 from Rapid Electronics.

The capacitor value was chosen based on the resistor value and the desired attenuation. To provide 10x attenuation, the reactance of the capacitor needs to be one tenth of the resistance of the resistor. The capacitance required was calculated from this reactance and the already chosen resistor value using the standard equation for the reactance of the capacitor as shown in equations 5.2 to 5.4. A 22 mF capacitor was chosen as the lowest value, which when tolerances were taken into account, was guaranteed to be above the calculated value.

$$\frac{1}{2\pi fC} = \frac{1}{10}R \tag{5.2}$$

$$\frac{1}{100\pi C} = \frac{1}{10}2.2 \tag{5.3}$$

$$C = \frac{1}{22\pi} \approx 0.0145\,\mathrm{F} \tag{5.4}$$

Adding the power filtering board reduced the noise to a great extent and the problem of the noise floor on the medium gain channel being above the saturation point of the low gain channel was eliminated. The noise in the final system is discussed later in the testing section.

To reduce electromagnetic interference, the system was enclosed in an aluminium box. The box was designed with multiple compartments so that different boards were screened from each other and included a compartment in which the device under test could be enclosed, if needed. However, fitting and removing the lid had no noticeable impact on noise and as such, it was concluded that electromagnetic interference was not a significant source of noise. The aluminium box remains as a mechanical mount for the system.

A USB isolator board based on the ADuM4160 [9] and a board to power it were also designed and provision was made for these boards in the case. However, since the noise problems had already been reduced to manageable levels and it was believed the isolator could cause other problems (in particular it would limit the USB link to full speed rather than high speed), the USB isolator was never installed.

5.7 Calibration

To convert the raw ADC readings into current readings, scale factors and offsets are applied. In principle, these could be obtained by calculation from the circuit design and component values. However, in practice, imperfections in the system are likely to mean that calculated values are likely to be imprecise. Therefore, the scale factors and offsets are obtained through a calibration process. The calibration process involves taking readings both under known loads and with zero load applied. Calibration readings are averaged over a period of one second (100 000 samples) to reduce noise.

The calibration process uses two points for each channel. One point is measured with no load applied. The other point uses a test load. For each calibration measurement, the voltage delivered to the connections for the device under test is measured with a multimeter. The multimeter must then be removed before actually performing the calibration test so it does not influence the readings. The reading with no load applied directly determines the offset. The scale factor is determined from both calibration readings by Equation 5.5 where S is the scale factor (in microamps per ADC unit), V is the voltage delivered to the test load (in volts), Z is the zero calibration reading (in ADC units), L is the loaded calibration reading (in ADC units) and R is the resistance of the test load (in ohms).

$$S = \frac{1\,000\,000\frac{V}{R}}{L - Z} \tag{5.5}$$

The scale factor and offset can be used to calculate current drawn by the sensor node from ADC readings using Equation 5.6 where I is the current (in microamps), A is the ADC reading (in ADC units) and other variables are as above.

$$I = S(A - Z) \tag{5.6}$$

Since the channels have very different gain, a separate test load was used for each channel. The high gain channel is calibrated with a 1 MΩ resistor as the load. The medium gain channel is calibrated with a 10 kΩ resistor as the load. The low gain channel is calibrated with a 100 Ω resistor as the load. All resistors are 1 % tolerance. These loads represent 30 % to 50 % of the full scale value for each channel depending on what voltage of device under test the system is set up for.

Calibration is undertaken at the start of each measurement session. As mentioned previously, a dialog box was added to the data acquisition application to collect the calibration measurements. These values are then stored in memory and written out in the header of the measurement file. To facilitate easy connection and disconnection of the resistors during the calibration process, calibration resistors loads are installed in a plastic box with 4mm connectors. A screen shot of the power measurement data acquisition application with the calibration dialog box visible can be see in Figure 5.7. The dialog box displays both the raw ADC values read during calibration and the resulting scale factors.

5.8 Data format

The acquisition application stores the measurements in a custom binary file format. A binary format was chosen over a text format to reduce the size and increase the speed of reading and writing. At the start of the file is a 16 byte "pre-header". The first 12 bytes of the pre-header are the string "WSNPWRMEASUR" to identify the format of the file. This is followed by the version and the header size as 16-bit little-endian integers. By specifying the header size in the pre-header, new fields can be added to the header without breaking compatibility with existing processing code.

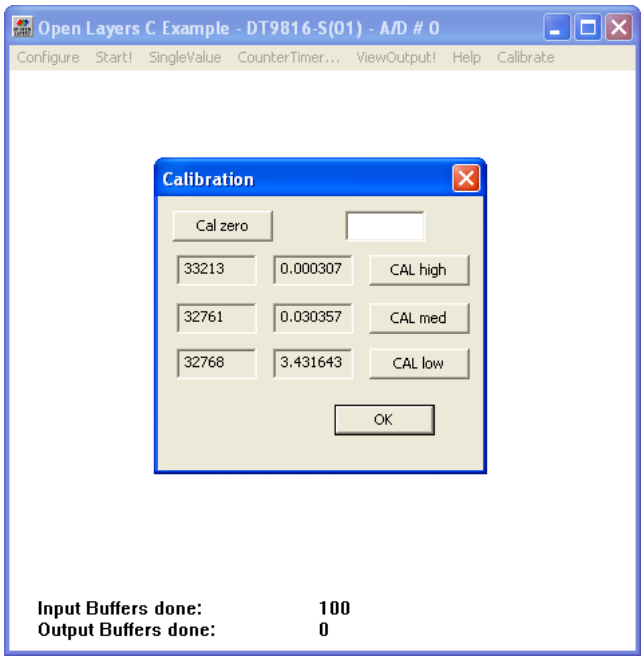


Figure 5.7: Screen shot of power measurement data acquisition application with calibration dialog visible

After the pre-header is the header. The header is divided into three sections. First are the global fields described in Table 5.2. This is followed by the channel specific header fields described in Table 5.3 which are repeated for each channel in order. Finally, a text description of the files contents (supplied by the user) is placed at the end of the header. If new fields are to be inserted in the header without breaking compatibility, they must be inserted between the current channel specific header fields and the description.

The type column in the tables uses BYTE for a single byte, WORD for a 16-bit integer, DWORD for a 32-bit integer and DOUBLE for a double precision floating point value. All fields are little-endian. After the header, the raw measurement data is stored with the values for the channels interleaved.

Offset	Type	Description	Notes
0x00	BYTE	Channel count	Always 3 with current system
0x01	BYTE	Data word size in bytes	Always 2 with current system
0x02	WORD	File offset at which description is stored	
0x04	DWORD	Total samples per channel	
0x08	DOUBLE	Voltage during "zero calibration"	Supplied by the user during calibration

Table 5.2: Global header structures for power measurement data format

Offset	Type	Description	Notes
0x00	WORD	Average ADC reading during "zero calibration"	
0x02	WORD	Average ADC reading during "loaded calibration" of this channel	
0x04	DWORD	Value of calibration resistor (in ohms)	currently hard coded in measurement collection application
0x08	DOUBLE	Voltage during "loaded calibration" of this channel.	Supplied by the user during calibration

Table 5.3: Channel specific header structures for power measurement data format

5.9 Data processing

To process the data into a usable form, a number of steps are performed in software. First, the signals are filtered and decimated to remove unwanted high frequency components and to reduce the size of the dataset to a more manageable level. Then the signals are scaled so that the values for the three channels represent current on a consistent scale of microamps. Finally, the signals are recombined. Initially a Matlab based solution is used for this processing but this was later replaced by a Java based one. The source code for both implementations can be found in Appendix F

The filtering is performed by convolving the signal with a FIR filter generated using signal wizard [102] with a cut-off at 5 kHz and 511 taps. The cut-off frequency for this filter was chosen based on the practical frequency performance of the system. The sampling rate is then reduced by a factor of 10 to 10 kHz (twice the cutoff frequency of the filter).

To convert the readings from the DT9816 to meaningful and consistent units (µA were chosen though other units could have equally well been used), the scale factors from the calibration process are used. As described previously, the actual conversion process is a simple case of subtraction followed by multiplication.

To recombine the signals, the following algorithm is used. If the value from the high gain channel is less than 8 µA then it is used. Otherwise, if the value from the medium gain channel is less than 800 µA, then it is used. Otherwise, the low gain channel is used. This algorithm effectively ignores channels that are in or near saturation.

In the initial processing code, the whole dataset was loaded into Matlab and converted to double precision floating point during the read process. It was then split into channels, filtered, decimated and recombined using routines supplied by Matlab. After processing the data remained in Matlab's memory, where it could be saved in any desired format using Matlab's built in features. This solution was quick to write and worked acceptably for small data files but it was problematic for large data files. In particular, immediately after filtering there were three complete copies of the data set in memory (the copy initially read in, the copy split into channels and the copy split into channels and then filtered) and each of these copies was in double precision floating point form. This meant that the memory usage of Matlab at this point in the process was over twelve times the size of the input file. For large

input files this led to significant swap activity and therefore very long processing times.

To speed up processing of large data files, a new processing application was written in Java. In this software conversion to floating point, filtering and decimation were combined into a single process and this process was performed in blocks so that only a small proportion of the raw data needed to be in memory at one time. This new design eliminated the need to keep large amounts of non-decimated data in memory at once. After the initial filtering and decimation, scaling and recombination were performed as before (though the different language meant that new implementations were needed). The system retains the data after filtering and decimation as well as the versions of the data after scaling and the final recombined data in memory. This data is made visible to the user as graphs using jFreeChart. The data can also be saved in a tab separated text file for further processing with other tools.

A "summary" feature was also added to the new software to extract and average blocks of similar readings even if noise made the transitions between those blocks impossible to see on a graph. This was added to make sense of the tail-end of the resistance box testing discussed later which was below the values that could be seen on a graph. The summary feature is controlled by two parameters, an "averaging length" (n) and a minimum size of block to keep (k).

To produce the summary, the processing program loops through the recombined data one sample at a time. For each sample, the program decides whether to add the current sample to the current block or to start a new block at the current sample. To make this determination, two mean averages are used. The first average (μ_c) represents the samples already in the current block while the second average (μ_n) represents the new sample.

The group of samples used to calculate μ_c begins at the start of the current block. If there are more than (n) samples already in the current block then the average is taken over the samples already in the current block. If there are fewer than n samples in the current block and there are sufficient samples remaining in the dataset to do so, then the average is taken over n samples. Otherwise, the average is taken to the end of the dataset.

The group of samples used to calculate μ_n starts at the current sample. If there are sufficient samples remaining in the dataset to do so then the average is taken over n samples. Otherwise ,the average is taken to the end of the dataset.

If $0.95 \leq \mu_n/\mu_c \leq 1.05$ then the sample is added to the current block. Otherwise, the existing block is completed and a new block started. If the completed block is larger than the minimum size, the start time, end time, number of samples and mean of samples in the block are recorded (note: for small blocks this is different from the μ_c used in determining block boundaries). Blocks shorter than the minimum size are not written to the output. A possible enhancement would be to add a feature that optionally combines sequential undersized blocks so that there are no gaps in the summarised data.

5.10 Complete system testing

Once the system was complete it was considered necessary to test two aspects of its performance. The first was whether it maintained the desired accuracy over a wide range of currents. The second was whether it had problems recovering from saturation. The former was tested by using resistance boxes to step through a range of loads. The latter was tested by using a resistor and transistor controlled by a DDS signal generator to apply a pulsed load.

Testing of the frequency response was not considered necessary as it was felt that it was highly unlikely that the instrumentation amplifiers or the data acquisition system would limit frequency response beyond the limitations already posed by the earlier stages of the signal chain which had already been tested.

5.10.1 Testing with resistance box

To check that reasonable accuracy was maintained over a wide range of input values, testing was performed using resistance boxes to gradually apply higher resistance loads. The test was run with the voltage at the node terminals set to 5 V. The test resistance started at 100 Ω and was increased in 100 Ω steps to 1 kΩ. It was then increased in steps of 1 kΩ to 10 kΩ, then steps of 10 kΩ to 100 kΩ, then steps of 100 kΩ to 1 MΩ and finally steps of 1 MΩ to 10 MΩ. The time spent at each step varied since the stepping was a manual process but was always at least a second.

Two different resistance boxes were used. The first was used for resistances up to 10 kΩ while the latter was used for resistances from 10 kΩ to 10 MΩ. The 10 kΩ test was performed with both resistance boxes. The gap where the first resistance box was disconnected and the second connected was clearly visible in graphs of the test.

At the time, the data was not analysed thoroughly, the graphs were just inspected visually and deemed to be sensible. Later, the data was analysed using the summary feature in the Java-based analysis software with a 500 point averaging length and a 300 point minimum group size. The results from processing the data with this summary feature are given in Tables 5.4 and 5.5.

The results from the test with the high resistance box made immediate sense and showed that with averaging, an accuracy of 2 % or better was maintained down to around 7 µA, and an accuracy of 10 % or better was maintained down to loads of half a microamp. This is well within the accuracy targets.

The results from the low resistance box were harder to analyse. Firstly, there was one more reading than expected and it was not immediately clear which reading was the extra one. After trying to analyse the data under multiple different assumptions, the author concluded that the extra reading was most likely caused by inadvertently making an 11 kΩ measurement at the end of the measurement sequence. The design of the resistance box would make this easy to do. Secondly readings with resistor values from 200 Ω to 6000 Ω were over-read by 7-16 % while the reading with a 100 Ω load was far closer to the calculated value. The author

resistance setting (Ω)	calculated current (µA)	measured current (µA)	error (%)
100	50000	50969	1.9
200	25000	26791	7.2
300	16667	18927	13.6
400	12500	14471	15.8
500	10000	11606	16.1
600	8333	9679	16.2
700	7143	8294	16.1
800	6250	7264	16.2
900	5556	6456	16.2
1000	5000	5808	16.2
2000	2500	2907	16.3
3000	1667	1938	16.3
4000	1250	1455	16.4
5000	1000	1164	16.4
6000	833	970	16.4
7000	714	723	1.2
8000	625	632	1.2
9000	556	562	1.1
10000	500	504	0.8
11000	455	458	0.9

Table 5.4: Results of resistance box testing from 100 Ω to 11 kΩ

resistance setting (Ω)	calculated current (µA)	measured current (µA)	error (%)
10000	500.00	505.34	1.1
20000	250.00	252.63	1.1
30000	166.67	168.52	1.1
40000	125.00	126.35	1.1
50000	100.00	100.97	1.0
60000	83.33	84.13	1.0
70000	71.43	71.98	0.8
80000	62.50	63.03	0.8
90000	55.56	55.98	0.8
100000	50.00	50.36	0.7
200000	25.00	25.10	0.4
300000	16.67	16.72	0.3
400000	12.50	12.53	0.2
500000	10.00	10.01	0.1
600000	8.33	8.36	0.3
700000	7.14	7.26	1.6
800000	6.25	6.38	2.1
900000	5.56	5.68	2.3
1000000	5.00	5.12	2.3
2000000	2.50	2.57	2.7
3000000	1.67	1.72	3.4
4000000	1.25	1.30	4.0
5000000	1.00	1.04	4.4
6000000	0.83	0.87	4.9
7000000	0.71	0.75	5.6
8000000	0.63	0.66	6.3
9000000	0.56	0.59	6.6
10000000	0.50	0.54	7.4

Table 5.5: Results of resistance box testing from 10 kΩ to 10 MΩ

believes this was because neither the calibration process done for the resistance box test[1] nor the resistance box test itself took account of voltage droop at high loads and was not a flaw in the measurement system itself.

5.10.2 Pulse testing

To test recovery from saturation, a pulsed test-load was used. Specifically, a 100 Ω resistor switched by a transistor was connected to the system. With the system set up to provide 3V to the load, this gave a current of 30 mA with the transistor on and drew no measurable current with the transistor switched off. The transistor was controlled by a chain of pulses from a TTI TG1010 DDS function generator [116] which was set up to generate a 10 ms pulse once per second. The graphs of the individual and combined data were then investigated for any signs of artefacts. The only artefact noticed was some ringing. This artefact is an inevitable result of linear phase FIR filtering. The ringing for several pulses on the medium and high gain channels was inspected visually. For the medium gain channel, the ringing when entering and leaving saturation was sufficiently small in magnitude that values seen during the ringing were well within the range that is considered "in saturation" by the recombination process. For the high gain channel, the ringing was sufficiently small in magnitude that while it was clear ringing from the digital filtering was present, it was not possible to visually distinguish between ringing and noise.

A graph of the data from after recombination can be seen in Figure 5.8 (note: the graph only shows the absolute value of the data and not the sign) While the noise shown on the graph is generally in the region of 5-6 µA and has occasional peaks to around 8 µA, as mentioned in the previous section, much smaller currents can be measured by the system when averaging over long periods. A graph focusing on a single pulse and showing both individual channels and the recombined data can be seen in Figure 5.9.

[1] The calibration process can take account of voltage droop when measuring current through the calibration resistor but only if the user makes a measurement of the reduced voltage and provides it to the system. This was not done for the resistance box test.

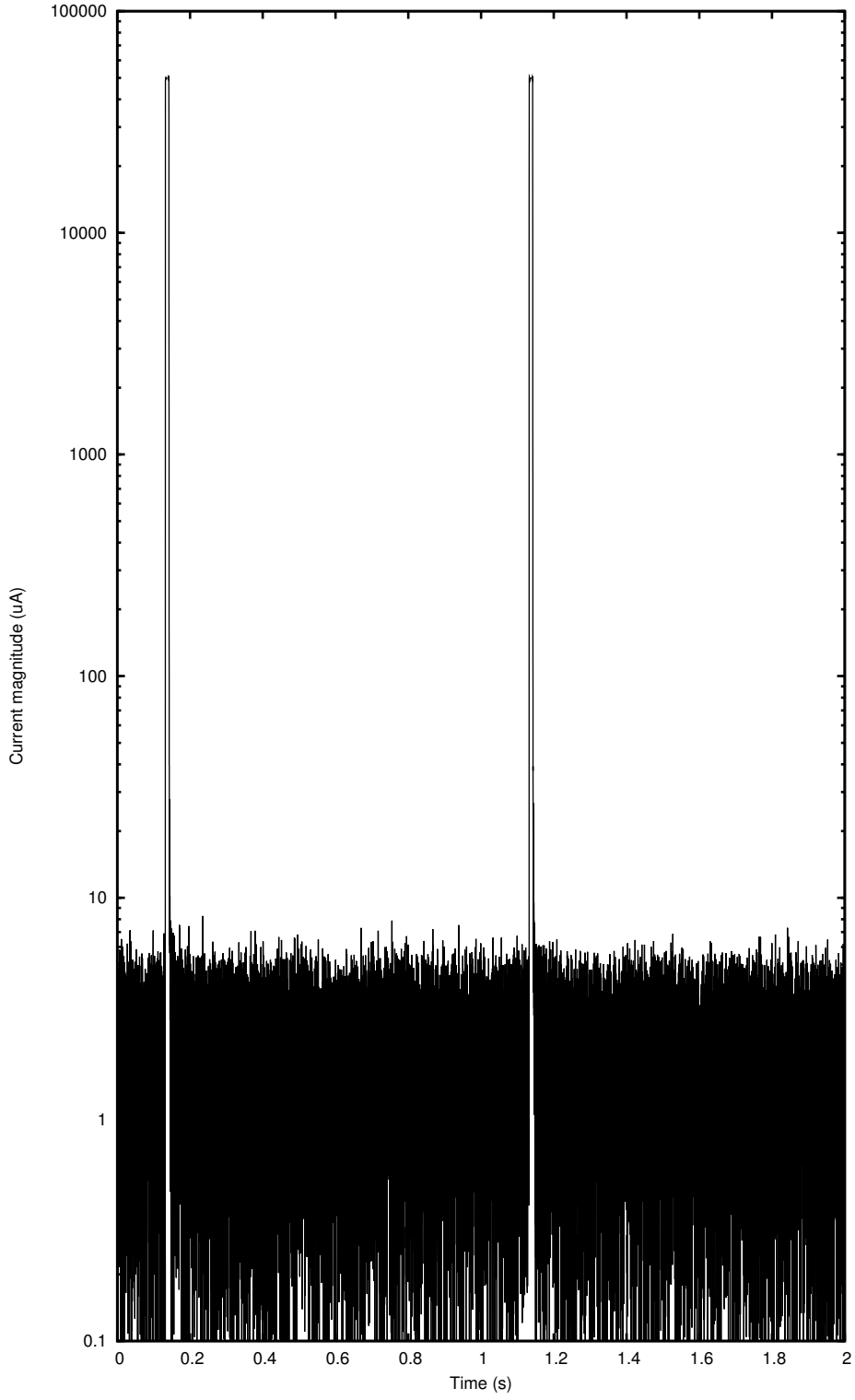


Figure 5.8: Combined data power measurement system during pulse testing

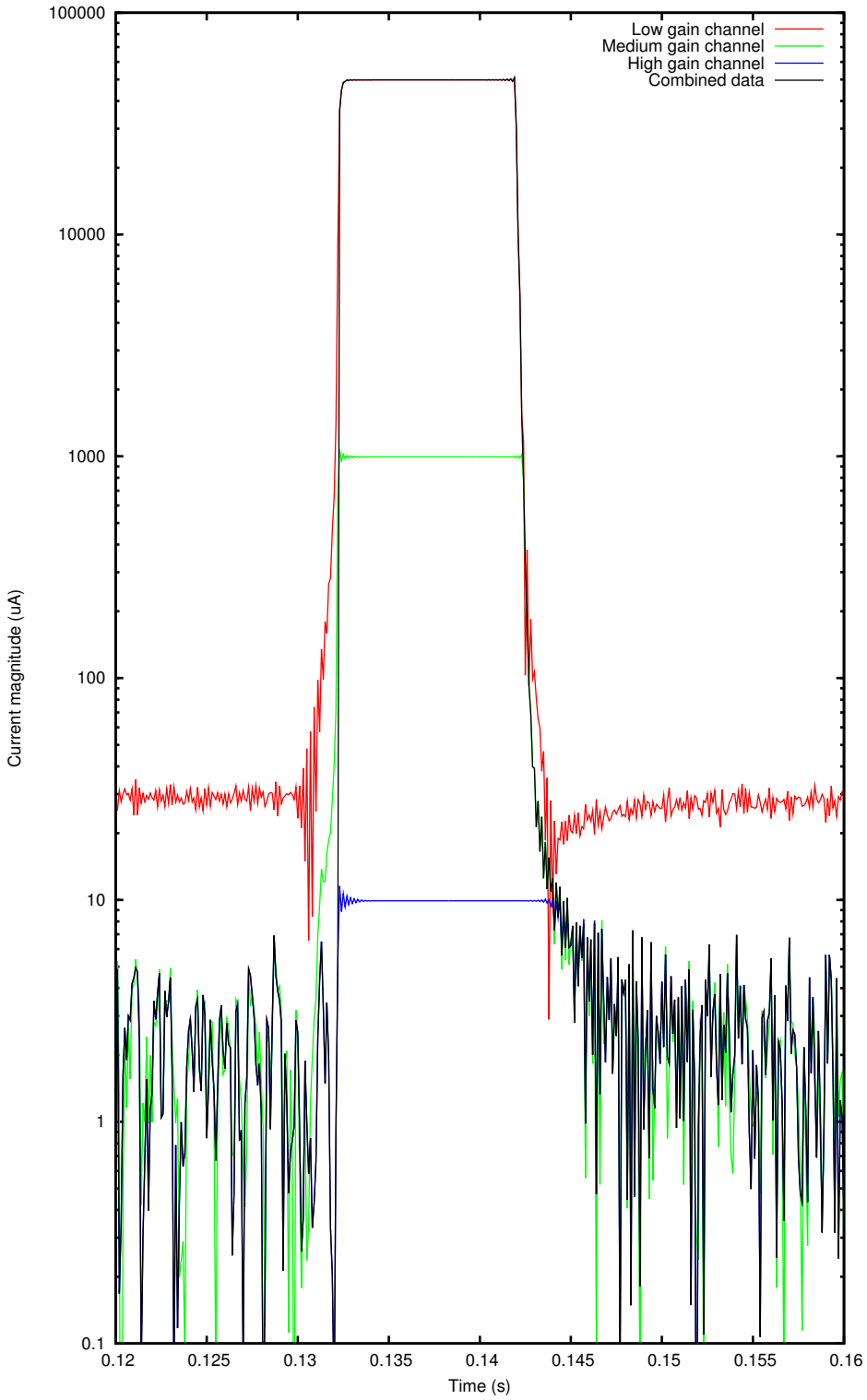


Figure 5.9: Output of power measurement system during a single pulse; data from low gain channel in red, data from medium gain channel in green, data from low gain channel in blue, combined data in black

5.11 Evaluation and potential improvements

It was disappointing that currents at the lower end of the desired range could only be measured through averaging; they could not be measured in individual sample points or visually determined from graphs of the data due to noise. However, this was not felt likely to be a significant problem for actually characterising sensor node power consumption as it was felt unlikely that a node would transition between different low power states without having a high power state in-between.

The frequency response was less than originally desired and had the cause of this been determined earlier, components may have been changed to extend the frequency response. Such changes may be made to the system later if it is deemed useful for future experiments but there are no immediate plans to do so. Modifications to extend the frequency response would be complicated by the fact that the filtering and decimation process was designed after realising the more limited frequency response of the hardware but before realising the root cause of that more limited frequency response. So to get the full originally desired frequency response, it would be necessary to change both the hardware and the processing software.

USB isolation was not pursued since it was felt it could interfere with fast and stable USB operation and the noise had already been reduced to acceptable levels. Since it was not pursued, it is not known what, if any, impact it would have on noise levels.

Overall, the system was deemed sufficient to serve the purpose of evaluating and checking the power consumption of sensor node hardware.

Chapter 6

RF measurement systems

An experimental set-up was designed for taking measurements of radio propagation. The primary goal of this set-up was to allow measurements of the attenuation exhibited by a radio link through a field of crops, ideally to within a few dB of the actual value and with sufficient sensitivity to measure any channel that could be used for wireless communication with off-the-shelf wireless hardware. An obvious approach was to use a portable vector network analyser. However, vector network analysers tend to have noise floors that are too high to allow direct measurement of such channels. For example, according to its maintenance guide, the Anritsu MS2036A operating in VNA mode between 10 MHz and 3 GHz, is supposed to have a noise floor of below -80 dB. Tests with our MS2036A showed that its noise floor was generally just below -80 dB though there were occasional peaks above this value. For comparison, the MRF24J40 [68] has a nominal transmit power of 0 dBm, and a nominal receive sensitivity of -90 dBm meaning that in principle, it should work with a channel attenuation of up to 90 dB. Adding range extenders like the CC2591 could increase the tolerable channel attenuation further to around 120 dB[1].

A vector network analyser could be combined with one or two external amplifiers (a low noise amplifier on the input, a power-amplifier on the output or both) to measure higher attenuations but this would be complex to calibrate, would still leave the transmit and receive grounds coupled and could potentially leave a path for stray signals to go from transmitter to receiver bypassing the channel under test. Chirp sounding [97] seems like an interesting approach but the author was not aware of chirp sounding at the time of designing this system and has not investigated it in detail.

The solution adopted was to use a sine wave source and a spectrum analyser. The spectrum analyser is capable of detecting and measuring very small signals without external amplification and since two separate instruments are used, complete electrical separation between transmitter and receiver is possible. The signal source used was an Anritsu MS2036A portable vector network analyser [11] while the spectrum analyser used was a Rhode and Schwartz FSH8 [93].

Control of the instruments and storage of the measurements was performed by a laptop computer. To ensure complete electrical separation between the transmit and receive sides

[1] It was later discovered that this means of calculating tolerable attenuation was highly optimistic. Details are given in Section 7.4

of the system, an optical fibre link was used to connect the spectrum analyser to the rest of the system. This approach can be implemented using standard RF test equipment with a minimal amount of custom code to link it together. The system measures the propagation loss between the output of the source and the input of the spectrum analyser. The antennas are considered to be part of the channel being measured but nevertheless suitable antennas and stands for them had to be built to use the system. A block diagram of the field measurement system can be seen in Figure 6.1.

To calibrate for cable losses and for the fact that the actual output power of the device acting as a signal source was slightly different (a few dB) from its nominal output power, the source and spectrum analyser are connected together using the same cables that are used during measurement and a set of readings is taken in the same way as for field measurements. These calibration measurements are later subtracted from the measurements taken in the field. The antennas are considered to be part of the channel being measured.

The system makes a number of assumptions. Firstly, it is assumed that at a given frequency the signal source will output a signal of consistent level. Secondly, it is assumed that the spectrum analyser will produce accurate measurements across a wide range of power levels. The author believes that with quality RF test equipment, these assumptions are reasonable. It is also assumed that the highest peak on the spectrum analyser trace represents the signal generated by the source; potentially another strong source in close proximity could violate this assumption but it is felt this is unlikely in practice, especially given the narrow bandwidth used.

Unlike with the power measurement system, there was no custom circuitry in the signal path that could impact performance. Therefore, the performance of the field measurement system was determined by that of the instruments involved. The accuracy with which the loss is measured is determined primarily by the accuracy of the spectrum analyser but unfortunately, there was no obvious method of measuring this. It would be possible to attempt to compare the spectrum analyser used against another spectrum analyser or other piece of test equipment but it was not felt this would achieve much (all the other suitable equipment available was a lot further from their manufacturers calibration than the one used in this system). The maximum loss that can be measured is determined by the noise floor of the spectrum analyser which is around -140 dBm with the settings used here and the power output of the source which is nominally 0 dB (and in practice slightly higher).

A similar system was used to test the power amplifier in the Texas Instruments CC2591 range extenders [112]. Unlike with measurements on a passive system, testing a power amplifier requires control over the source power as the gain and harmonic output will vary with the input power. In this variant, the signal source was replaced with an Agilent E4438C [4] RF signal generator that was less portable but allowed control of the power; the fibre link was not used and the software was modified to control this new signal source and to measure signals at twice and three times the frequency of the source. A block diagram of the power amplifier measurement system can be seen in Figure 6.2. The system used for power amplifier measurement is considered a variant of the system used for field propagation measurements. The following sections will generally describe the field propagation measurement system first with changes made for the power amplifier measurement system explained at the end of each section.

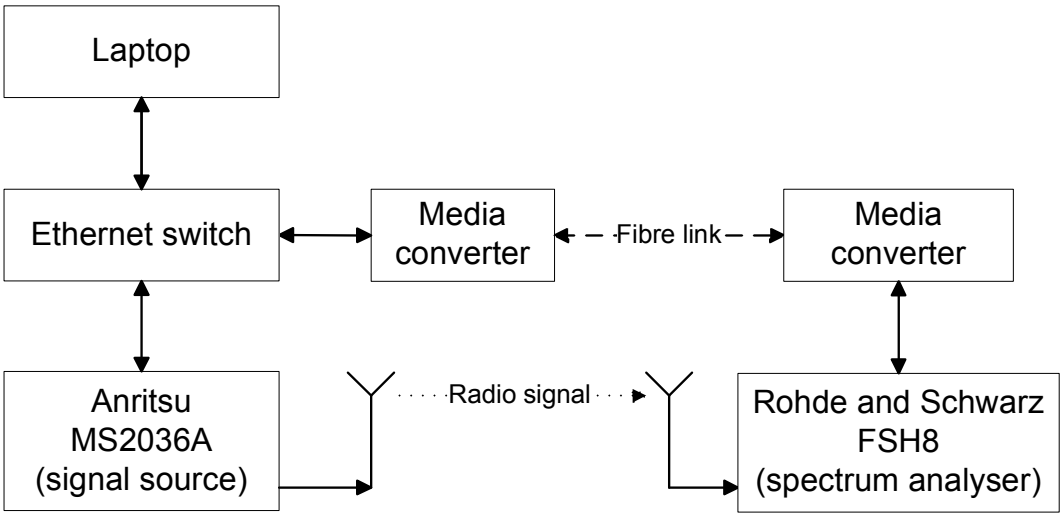


Figure 6.1: Block diagram of field measurement system

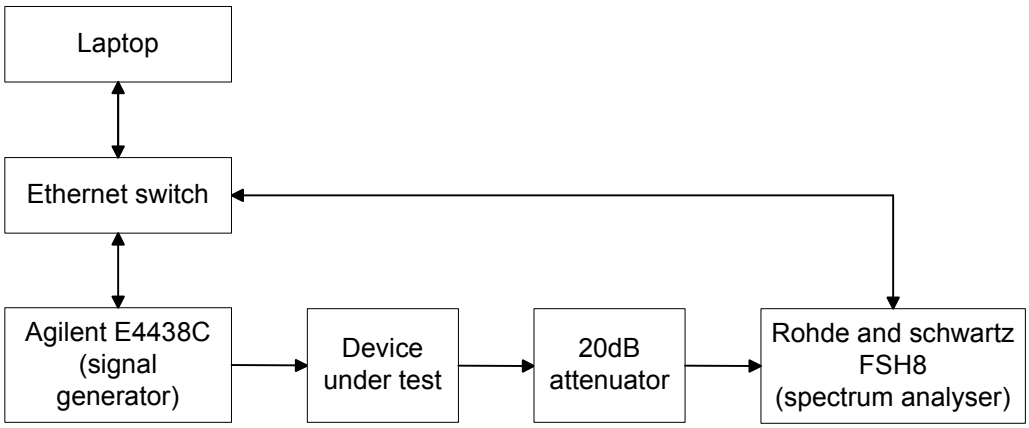


Figure 6.2: Block diagram of power amplifier measurement system

6.1 Hardware setup

As mentioned previously, an Anritsu MS2036A [11] portable vector network analyser was used as a signal source. While intended to be a vector network analyser, it is possible to make this device act as a remote controllable unmediated source by setting the span to zero and the sweep resolution to the minimum possible value. To receive the test signal, a Rhode and Schwartz FSH8 [93] portable spectrum analyser was used. The MS2036A and FSH8 were controlled over Ethernet from a laptop which was located close to the signal source.

To isolate the spectrum analyser from the rest of the system, a fibre optic link was constructed. The fibre optic link consists of two StarTech.com MCM110SC2GB media converters and a custom made reel containing 100 m of four core 62.5 µm multi-mode fibre. Only

two of the cores in the reel are actually used; the remaining two cores are there because four-core was the smallest core count available from the supplier.

The laptop, MS2036A and FSH8 were powered from their own internal batteries which were recharged as needed using a DC-DC converter connected to a sealed lead acid battery. The lead acid battery was only connected to one piece of equipment at a time thereby preserving electrical separation. The network equipment at the laptop/transmitter end was powered from the laptop, while the media converter at the spectrum analyser end was powered off a set of alkaline batteries. The laptop and the network gear at the laptop end were mounted in a hard case for protection and to keep the items together. A photograph of this case can be seen in Figure 6.3.

The system can operate over the frequency range 610 kHz to 6 GHz with the limiting factor being the frequency range of the Anritsu MS2036A used as as a signal source. However, performance has only been characterised in the 2.4-2.5 GHz band as that is where we were using the system. The Rohde and Schwarz FSH8 has a noise floor of about -140 dBm in the mode we are operating it in.

The power amplifier measurement system used a slightly different set of equipment. The signal source was changed to an Agilent E4438C [4] signal generator. Unlike the MS2036A this device provides control of the output power. A 20 dB attenuator was also introduced to protect the FSH8 from the output of the power amplifier. Finally the fibre optic link was deemed unnecessary due to the high signal levels and the fact that grounds would be coupled anyway and as such it was not used. The laptop, network switch and FSH8 were retained.

6.2 Measurement process

To measure propagation at a given frequency, first the signal source is set to the desired frequency. Then the spectrum analyser is used to measure a span centred on the transmit frequency. It is necessary to measure a range, as there is no way to lock the frequency sources in the instruments together. The span was set to 2 kHz based on the experimentally observed deviation between the instruments. It was felt that going any narrower would not reliably ensure that the signal source was within the range. The span was later widened to 3 kHz due to problems operating in cold conditions.

The FSH8 automatically selected a resolution bandwidth (RBW) and video bandwidth (VBW) of 30 Hz for the selected span and these default values were used (though the software sets them explicitly). It is assumed that the signal from the MS2036A is the strongest signal in the frequency range measured by the FSH8, which is a reasonable assumption given that narrow band point sources in the 2.4 GHz band are unusual. Therefore, finding the signal is a simple matter of finding the peak in the dataset returned from the FSH8.

The MS2036A used as a signal source has a nominal output level of 0 dBm. In the mode used here (preamp on, attenuation off, RBW 30 Hz, VBW 30 Hz) the FSH8 has a noise floor below -140 dBm. This allows measurement of link losses greater than most wireless sensor network hardware is likely to tolerate even with the use of range extender ICs. Measurements of

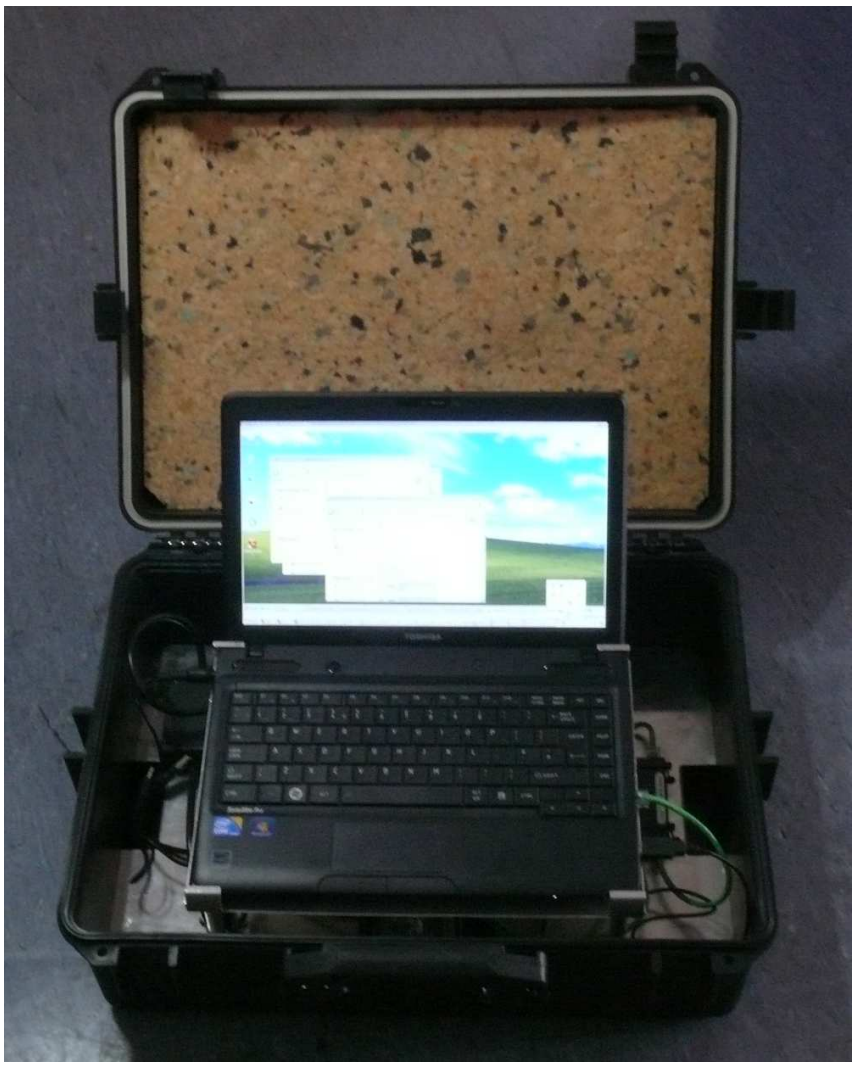


Figure 6.3: Laptop and associated network gear cased up for field work

maximum tolerable losses for wireless sensor network transceivers are documented in Section 7.4.

If the MS2036A actually transmitted at its nominal output power and all the cables were lossless then the peak measurements from the spectrum analyser in dBm would correspond to the loss of the link (including the antennas) in dB. In practice, the MS2036A transmits at a higher power than nominal and the cables are not lossless. While the impact of these two imperfections largely cancel out, it was still considered prudent to take a set of calibration readings (by directly connecting the transmit and receive antenna connectors with a male-male adaptor) and subtract the calibration measurements from the field measurements in post-processing.

The software on the laptop automatically sets up the instruments then loops through the process of setting the frequency of the signal source and then taking a reading using the

spectrum analyser. The spectral measurements are saved for later post-processing. The software was written using visual C++ with NI Visa [77]. The system uses a blocking approach and waits for each instrument to report that it has completed an action before moving on to the next action.

For the power amplifier measurements, the software was modified to work with the changed source and extended to take measurements at the second and third harmonic of the source frequency and to take measurements at multiple source powers.

6.3 Antennas and stands

There are two approaches to antennas for measurements like this. One is to use well characterised test antennas that have little in the way of interaction with their environment. This makes it easier for the effect of antennas to be eliminated from the results but it is unclear if doing so is actually desirable since the interaction of antennas with the environment is a major part of system operation in cluttered environments. The other approach is to use antennas of roughly the same type as are planned to be used on the actual nodes in the network. Antenna stands were developed to facilitate both approaches but only the latter approach was actually used in the project's fieldwork.

To allow mounting the antennas in a manner similar to how they will be mounted in an actual network, three different types of antenna stands have been constructed. Two copies of each type were produced to allow for symmetrical measurements.

The first to be constructed was a tube intended to be buried in the same way as a sensor node. This can be seen in Figure 6.4. This would provide the most realistic representation of a sensor node's antenna mount but was not used due to the difficulty of deploying it.

The second stand is designed to sit on the ground and mount an antenna vertically. This places the antenna further from the ground than it would be on a buried sensor node but is much easier to deploy. This stand was used for all low level measurements in the project's fieldwork and is pictured along with a hastily constructed coax monopole antenna in Figure 6.5.

The final stand is a set of telescopic tubes with weights at the bottom to allow mounting of antennas at suitable heights for crop canopy measurements. There are two different lengths of outer tube and three sizes of inner tube allowing an overall height range from 0.5 m to 2 m. These stands were used for the high level measurements in the 2012 fieldwork. A photograph of one of these stands can be seen in Figure 6.6.

The stands were all designed and constructed in-house with the assistance of the University's mechanical workshop.

Figure 6.4: Buried antenna stand for field measurement

Figure 6.5: Surface antenna stand for field measurement

Figure 6.6: Telescopic stand for field measurement

6.4 Measurement data format

Data from the measurement set ups described in the previous two sections is stored in a custom binary file. For each spectrum analyser result set, a header is stored containing details of the settings used. All fields in the header are 32-bit little-endian integers. After the header, the actual spectrum data read from the spectrum analyser is stored as a series of little-endian single precision IEEE 754 floating point values.

There are two versions of the header, as when developing the amplifier measurement set up it became necessary to store more information about each measurement. The version 0 header is 8 bytes long while the version 1 header is 24 bytes long. Details of the structure are shown in Table 6.1 while the units and other important information for each field are shown in Table 6.2. When reading a version 0 header, the nominal transmit power is assumed to be 0 dBm, the harmonic number is assumed to be 1 and the frequency span of the spectrum analyser is assumed to be 2 kHz.

The change in meaning of the first field from transmit frequency to version, limits the value of the transmit frequency field in a version 0 file to 0x00FFFFFF (otherwise the reading software will consider the file to be a version 1 file). This value represents a transmit frequency of over 16 GHz which is beyond the limits of the test equipment in use.

The post-processing software assumes that the records in the file will be ordered, first by frequency, then by transmit power and finally by harmonic number.

Offset	Version 0	Version 1
0x00	Transmit frequency	Version
0x08	Number of data bytes	Transmit frequency
0x0C		Nominal transmit power
0x10		harmonic number
0x14		frequency span of spectrum analyser
0x18		Number of data bytes

Table 6.1: Header structures for measurement data format

Field	Details
Transmit frequency	Measured in kHz
Version	value of 0x01000000 for version 1 format higher values will be used for future format versions (if any)
Nominal transmit power	Measured in hundredths of a dBm
Harmonic number	1=fundamental, 2=2nd harmonic 3=3rd harmonic etc
Frequency span of spectrum analyser	Measured in Hz
Data byte count	Four times number of samples in spectrum analyser sweep

Table 6.2: Field details for measurement data format

6.5 Post-processing

To view and process the data from the propagation and amplifier measurement systems, a custom application has been written. The application allows viewing of the individual spectrum readings taken by the measurement system but its main purpose is to summarise that data, subtract a calibration dataset if desired and then plot it against frequency, power or harmonic number.

For each spectrum analyser sweep, the parameter of interest is the magnitude of the single tone signal received by the spectrum analyser. Assuming that the single tone signal in question is the strongest source in the measured spectrum, its level will be represented by the peak of the spectrum analyser sweep. The post-processing software finds the peak of each sweep as it loads the data file into memory.

Optionally, a calibration dataset can also be read and processed in the same way. The results from this can then be subtracted from those in the main dataset to translate the received power measured by the spectrum analyser into the propagation gain or loss. Only the fundamental frequency information in the calibration dataset is used. Readings for harmonics in the output dataset are calibrated using the corresponding fundamental readings of the calibration dataset. It is possible to load the same dataset as both main and calibration datasets. Doing so allows the user to view the level of harmonics relative to that of the fundamental.

Three lists on the right hand side of the application window allow the user to select the frequency, power and/or harmonic they are interested in. To produce a summarised graph, the user selects a value in two of the three lists and a summary option (calibrated or uncalibrated) in the third. The selected data is graphed in the axis in the right of the application window and can be exported to a CSV file. Buttons at the bottom of the application window allow loading of a main dataset, loading of a calibration dataset, merging a new dataset with the main dataset to allow replacing failed measurements without redoing the whole measurement session, saving the main dataset and exporting the data behind the current graph. A screenshot of the application can be seen in Figure 6.7.

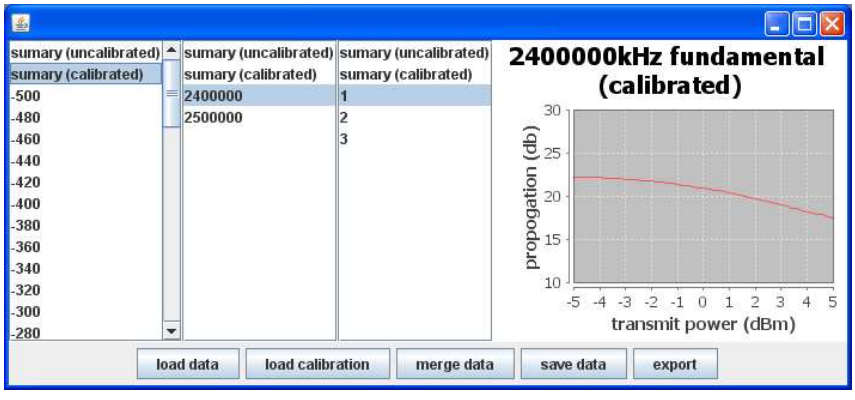


Figure 6.7: Screenshot of the measurement processing application showing how the gain of a CC2591 range extender varies with input power

6.6 Outstanding issues

While the set-up generally operates correctly, occasionally the spectrum analyser trace shows only noise. It appears that the technique used to set the MS2036A as a single frequency source occasionally fails to produce any output signal for no apparent reason. The settings displayed on the MS2036A do not appear any different on failed measurements than on successful ones.

This issue has been worked around by adding a threshold test to the measurement software. If the peak in the spectrum analyser results is less than -130 dBm then the measurement is assumed to be entirely noise and the measurement is repeated. After four attempts at making a given measurement without success, the software gives up and reports an error to the user. This solved the problem of missing data points but if duplicating this system, the author would advise finding a different signal source that does not suffer from the issue.

Also, the time taken to gather each spectrum analyser sweep (of which many are desirable for a measurement set) is around half a minute leading to a time of about 25 minutes for a 51 point sweep or 50 minutes for a 101 point sweep. The main contributor to the slowness is the time taken to obtain a spectrum analyser trace. This slowness is due to the FSH8 being a conventional spectrum analyser that processes the signal in the analogue domain and then digitises it. A modern digital signal analyser can provide comparable results far quicker because it can reuse the same input data for all the frequency points in the trace. Unfortunately, such digital signal analysers do not currently seem to be available in a form that is readily field-portable.

Chapter 7

Sensor node investigation and design

To gain practical deployment experience with a wireless sensor network in rapeseed a test platform is needed. The backplane system developed for the Wines 2 WSN4IP system(see Appendix A and research paper in Appendix B) was considered briefly but was considered unsuitable in many ways. Therefore other platforms had to be investigated.

Sensor nodes need to consist of a number of components. At a minimum there needs to be:

- A microcontroller to coordinate the operation of the node and its relationship with the network.
- A RF transceiver to convert the data from the microcontroller into a form that can be transmitted and convert the incoming signals back into digital data for the microcontroller.
- An antenna to transmit and receive radio signals.
- One or more sensors to make actual measurements.
- Batteries to provide power.

On prototypes there may be other Input and Output (IO) such as LEDs or a serial connection to assist in troubleshooting the network. A block diagram of a typical sensor node is shown in Figure 7.1

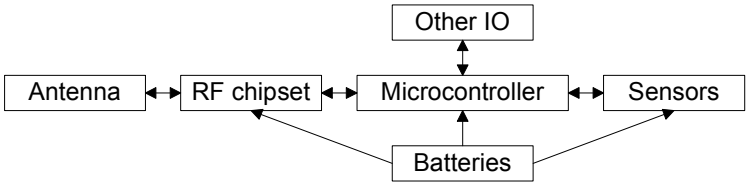


Figure 7.1: Block diagram of a typical sensor node

A sensor network must also have a base-station. The base station's job is to coordinate the network and to collect the data from sensor nodes. The collected data may be either stored, used to trigger actuators or forwarded over a longer distance network such as a mobile phone network. A base station needs to consist of the following

- A radio system for communicating with the sensor nodes. For practical reasons it is easiest to base this around the same microcontroller and RF hardware as the rest of the network.
- A system that is capable of providing capacity for more storage and software that can interface with the long distance network, such as an embedded Linux board.
- An interface for the long distance network such as a USB mobile broadband stick.
- Antennas both for communicating with the sensor nodes and communication over the longer distance network.
- Possibly a range extender device
- A suitably high capacity battery set-up to power the above equipment.

The network may also have repeater nodes. These are likely to have similar hardware to the regular nodes but may have larger batteries, range extenders or different antenna configurations.

7.1 Devices investigated

7.1.1 eZ430-RF2500

The first system investigated was the Texas instruments MSP430 eZ430-RF2500 Development tool [113]. This is a system consisting of two boards, one board contains a programmer and USB to serial interface while the other board contains the microprocessor and transceiver. There was also a battery holder which could be connected instead of the USB interface. This had the advantage of being physically small and at the time Texas Instruments claimed they were the lowest power solution available. However there were problems programming them (this may well have been a case of poor documentation) and the hardware had little in the way of IO available which was also a concern. Therefore this line of investigation was abandoned.

7.1.2 Picdem Z with MRF24J40

Next Microchip's PICDEM Z [69] was tried along with its MRF24J40 RF modules. This hardware had the advantages of a familiar development environment and the 40 pin microcontroller had large amounts of IO. However the PICDEM Z motherboard was impractically large, relatively expensive and very little of the IO was connected to switches or LEDs that could be used for troubleshooting.

Microchip offers two RF modules designed to be used with the PICDEM Z motherboard. Both are based on the same RF transceiver IC, work with the same software and have the same pin-out. The smaller and cheaper of the two modules is referred to by Microchip as the "MRF24J40MA PICDEM Z 2.4GHz RF Board". This module consists of the MRF24J40MA radio transceiver module mounted on a small adaptor board that converts it to the connector used on the PICDEM Z. The MRF24J40MA radio transceiver module has a built in PCB antenna and has no provision for use with an external antenna. The more expensive module is referred to by Microchip as the "PICDEM Z 2.4 GHz Daughter Card" and consists of only one board. It is physically larger, has provision for an optional external antenna connection (though the signal path to this connector appears to be missing part of the recommended balun circuit) as well as an extra edge connector (which allows its connection to the Microchip Explorer 16 if desired).

Microchip offers two different software stacks for use with their RF modules. One is a Zigbee stack, the other uses a Microchip specific protocol called MiWi. Zigbee has the advantage of being an industry standard but is more complex and the Microchip Zigbee stack has more onerous licensing restriction than the MiWi stack. In particular according to Microchip if you want to market a product based on the Zigbee stack you have to join the Zigbee alliance. Both of the Microchip stacks have licensing restrictions which allow their use only with Microchip micro controllers and RF transceivers. Based on these considerations the MiWi stack was chosen. Both stacks use IEEE 802.15.4 [46] as their physical layer.

MiWi supports both peer to peer and hierarchical networks. In the hierarchical configuration there is a root node known as the "PAN coordinator" [1] . Other coordinators connect to the PAN coordinator and end devices can connect to any coordinator. Nodes are usually addressed by "short addresses" which identify the node by which coordinator it is attached to. The PAN coordinator always has a short address of 0. In a wireless sensor network application it is likely that the node forming part of the field gateway would be set up to act as the PAN coordinator. In this way it will have a known address of 0 making it easy for the nodes to send reports to it.

MiWi end devices can be either "full function" or "reduced function". Full function devices have their radios listening all the time while reduced function end devices do not. Packets sent to reduced function end devices are queued in the associated coordinator node.

7.2 Development of node hardware based on the MRF24J40

The Microchip system seemed to offer an attractive combination of well known software and suitable hardware but the PICDEM Z motherboard was both expensive and very physically large and as such custom board options were considered.

The first step taken was to keep the Microchip RF modules but mount them on a board based on the microcontroller board that the School uses for teaching microcontroller programming to undergraduates. Basing it on this board allows it to be either used standalone

[1] PAN is an acronym for "personal area network", although the network under development here is not a personal area network the 802.15.4 standard was designed for PANs and hence the term is PAN used in systems built on it

or (for applications where more troubleshooting/experimenting IO is desirable) for it to be combined with the IO board that the School uses to go with their microcontroller board. This custom microcontroller board will hereafter be referred to as the "RF micro board". Two revisions of this board were produced, between the first and second revisions the footprint for the serial level shift IC was corrected (the original was too wide making construction very difficult) and disconnect jumpers were added to the power feeds to the microcontroller, serial level translator and RF module. This would have allowed them to be measured separately with the power measurement system and will also allow the serial level translator to be isolated to save power. Both types of RF module mentioned in the previous section have been successfully tested with the RF micro board. Figures 7.2 and 7.3 show the RF micro board with the two different RF modules fitted.

However while it was a useful board for gaining more experience with MiWi the RF micro board was still too large and physically unsuitable for designing into a sensor node. Further an external antenna was considered desirable so the node could be buried and the antenna could be above ground and the only MRF24J40 module available at the time with an external antenna connection was expensive, physically inconvenient and appeared to be lacking parts of the balun circuit from the ICs data sheet [2]. It was also desirable to have a real-time clock to control the node's wake-up cycle.

Based on these considerations, it was decided to change microcontroller from the PIC18F4620 to the PIC18F45J11 [72] as the PIC18F45J11 has an internal real time clock. It was further decided to design a custom PCB that would directly accommodate the MRF24J40 without using a module and that would be suitable for integrating into the node.

It was determined that integrating the RF chip onto the custom board would require a 4-layer board and professional assembly. Therefore it was decided to design the PIC based section of the board first, build a version of it with a RF module connector and ensure that all the PIC based section of the circuit and the IO assignments used to communicate between the new PIC and the RF transceiver were workable. After testing this board and fixing a design problem regarding the interrupt line from the transceiver, the final hardware complete with RF transceiver chip was designed and sent for fabrication and assembly.

The board was designed to provide measurement of soil resistance, detection of surface water and temperature measurements. Soil resistance is measured through probes in the ground which are excited with an AC signal so the readings are not biased by DC currents already in the ground. Water presence is detected through contacts on the top of the node. Temperature is measured using two MCP9701A temperature sensors [70]. One of the temperature sensors is mounted deep in the node to give a representation of ground temperature and the other is mounted on the external antenna connector to give a representation of air temperature. All sensing hardware is designed so it can be powered down while the node is asleep.

Having designed, tested and debugged the PIC based section of the board, the final node hardware board with the RF transceiver chip onboard was designed. A picture of this board can be seen in Figure 7.4 and the schematics can be found in Appendix P.

[2] Since this work was undertaken Microchip have released the MRF24J40MC module [71] which has both an external antenna connection and an on board range extender

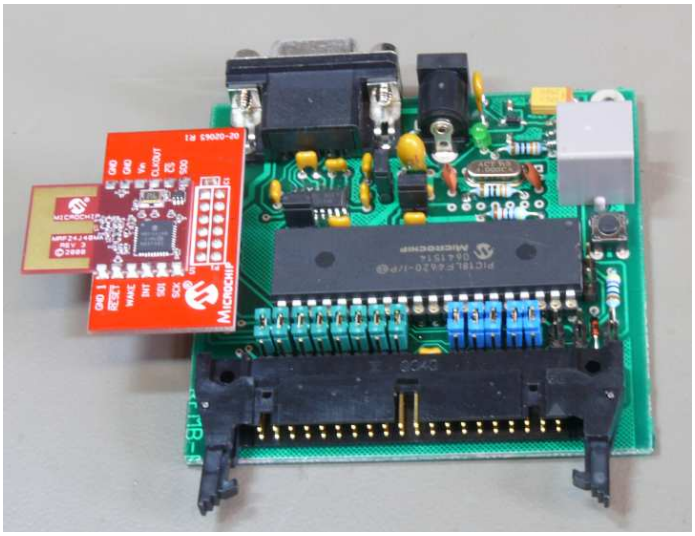


Figure 7.2: RF micro board with MRF24J40MA PICDEM Z 2.4GHz RF Board

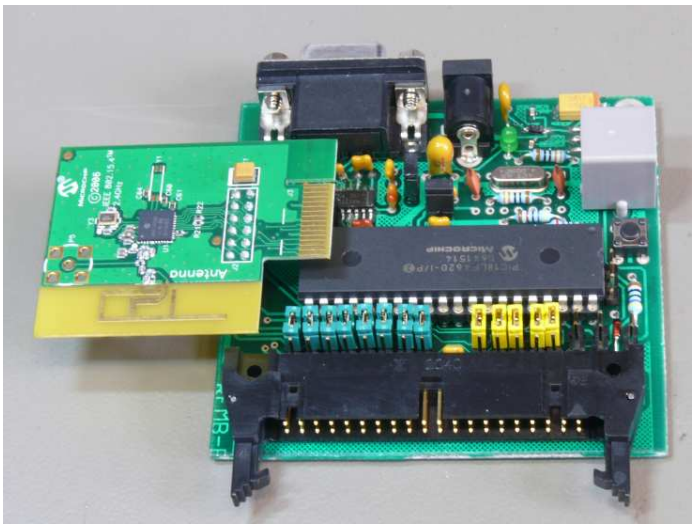


Figure 7.3: RF micro board with PICDEM Z 2.4 GHz Daughter Card

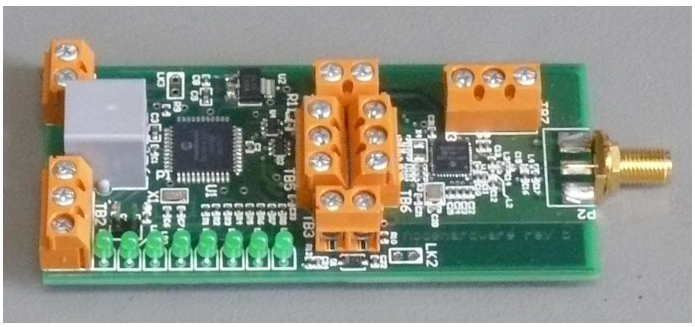


Figure 7.4: Node hardware board

7.3 Investigation and use of range extenders

During the field trip described in Section 8.1 it became clear that path losses in crops were very high. As a result consideration was given to the use of "range extender" ICs. These ICs combine a low noise amplifier, a power amplifier and signal switching systems.

This decision to use range extenders was made too late to integrate the range extenders into the main node hardware. Fortunately the required signal from the RF transceiver had been brought out as a "just in case" measure and range extenders on a separate board were used in the base station and repeater nodes. The specific range extender being considered was the Texas Instruments CC2591 on the manufacturer's demo board as pictured in Figure 7.5.

The range extender board is connected to the node hardware board using the antenna connector on the node hardware board. It also requires a 3.3 V power supply, a direction signal and an enable signal. The direction signal came from the RF transceiver chip and had already been brought out to a connector. The enable signal was connected in parallel with the enable signal from the PIC to the RF transceiver and along with the power supply this had to be obtained by making wire-mods to the PCB. The ground connection was provided by the outer connection of the antenna connector.

It was also found that to achieve low power sleep a number of pull-down resistors had to be removed from the range extender board to prevent significant amounts of power being wasted in the resistors when these lines were high.

Tests were made on the power amplifier in the range extenders using the power amplifier measurement system described in Chapter 6. As described in that chapter the signal source was an Agilent E4438C [4] signal generator and the receiving device was a Rhode and Schwartz FSH8 [93] spectrum analyser. A 20 dB attenuator was used to protect the FSH8 from the output of the power amplifier. Tests were performed at frequencies of 2.4 GHz and 2.5 GHz. Measurements were taken of the fundamental and the second and third harmonics. Higher harmonics were not measured because of the limitations of the spectrum analyser.

Figure 7.6 shows the gain as a function of input power at 2.4 and 2.5 GHz. Figure 7.7 shows the level of the second and third harmonics relative to the level of the fundamental frequency. As expected as input power increases the gain decreases and the harmonic content increases.

Since this was only for a trial deployment in a location a long way from any possible interference victims no attempt was made to determine whether these harmonics needed to be suppressed. For a commercial product these harmonics would have to be investigated.

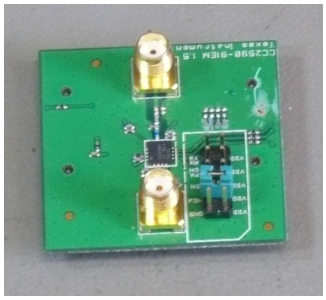


Figure 7.5: TI range extender demo board

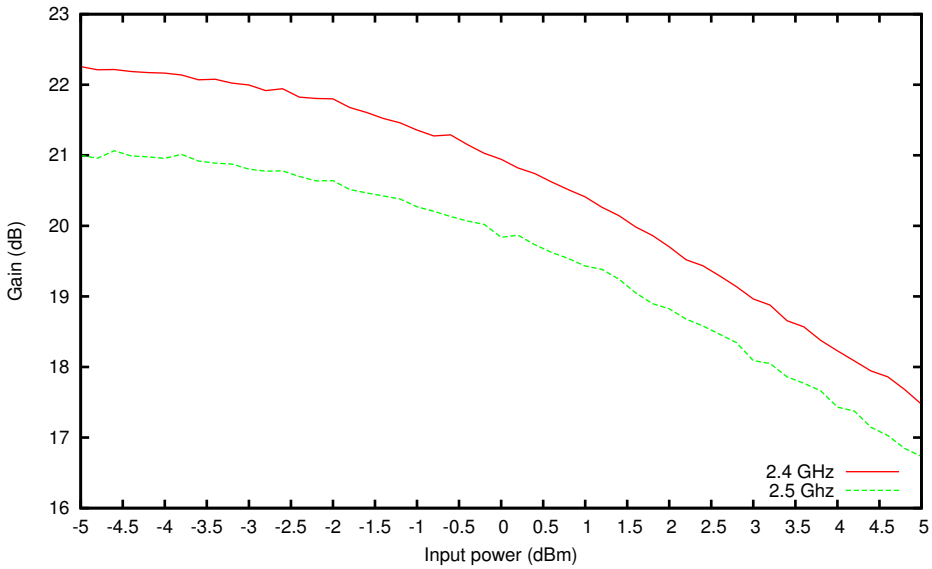


Figure 7.6: Gain of range extender verses input power

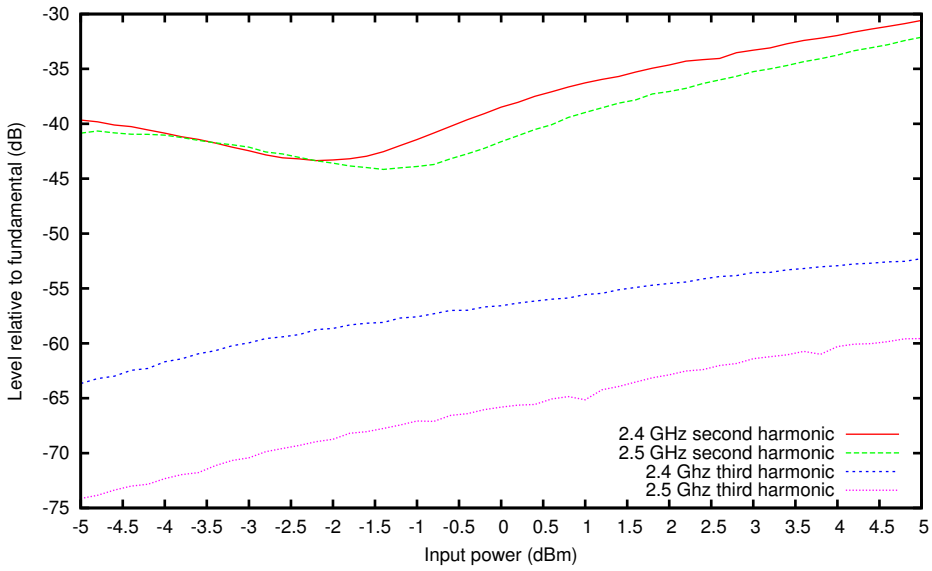


Figure 7.7: Harmonic content of range extender output verses input power.

7.4 Link budget measurement

It is not possible to measure the noise performance of the receive amplifiers in the RF transceivers directly since they are integrated into a single IC. Therefore it was decided that the most sensible method of determining the link budget was by direct measurement using a simulated channel made up of attenuators and a four port resistive splitter to allow monitoring.

One of the nodes was connected to the base station hardware as it would be in a real deployment. The other was battery powered and enclosed in a shielded cabinet to ensure that the simulated channel was the only communications path between the nodes. The shielded cabinet was needed as it has been observed that two nodes sitting together on a desk with no connections or antennas will communicate successfully just from the PCB traces radiating.

Activity in the channel was monitored by using one of the ports on the four port splitter to feed a Microchip ZENA packet sniffer. Additional attenuators were then inserted into the channel and the behaviour of the nodes was monitored over a period of time corresponding to at least 10 packet intervals (each packet interval being 10 seconds in the version of the software that was being used) and characterised as either "flawless communication" (all packets getting through) "reliable communication" (some packets having to be retried but all communication appeared successful), "poor communication" (some packets failing to get through even with retries) or "failed communication" (nodes never getting through the initial handshake). Of most interest for practical network deployments is where the performance goes from "reliable communication" to "poor communication".

For attenuation, mini circuits VAT+ series attenuators were used. A 30 dB attenuator was placed in the line to the base station, a 10 dB attenuator was placed into the line to the ZENA and a varying amount of attenuation was placed in the line to the node in the cabinet. The attenuation was initially added in steps of 10 dB to narrow down the range of interest, then 1 dB steps were used in the ranges of most interest to give a finer precision result. The resistive splitter was custom built for the experiment and is described in more detail in Appendix D.

Before each test, the simulated channel was measured to find the actual attenuation of the channel as a whole (including cables, attenuators and splitter) so that any inaccuracy in the attenuator values would be allowed for. Initially the MS2036 portable VNA was used for this measurement but this became impractical as the attenuation increased, so a signal generator and spectrum analyser were used instead.

With no range extenders "flawless communication" was observed up to an attenuation of 73.9 dB. "Reliable communication" was observed at attenuations from 74.9 dB to 75.8 dB. "Poor communication" was observed at attenuations from 77.0 dB to 86.9 dB and "failed communication" was observed at an attenuation of 88.2 dB.

With one range extender "flawless communication" was observed up to an attenuation of 94.1 dB. "Poor communication" was observed at attenuations from 95.6 dB to 96.4 dB and "Failed communication" was observed at an attenuation of 97.7 dB.

With two range extenders "flawless communication" was observed up to an attenuation of 94.5 dB. "Reliable communication" was observed at an attenuation of 103.6 dB, "flawless communication" was observed again at 105.5 dB. "Poor communication" was observed at attenuations from 106.2 dB to 122.4 dB and "failed communication" was observed at an attenuation of 124.9 dB.

The transition from "poor communication" to "failed communication" shows an improvement from the introduction of one range extender of around 10 dB (comparable to the LNA gain of the range extender) and a further improvement from the introduction of the second range extender of around 20 dB (comparable to the PA gain of the range extender). This shows that the LNA in the range extender must be far lower noise than the receive amplifier in the transceiver.

The transition from "flawless communication" to "reliable communication" seems to be rather fuzzy with a "flawless communication" result being seen at a higher attenuation than a "reliable communication result" in the test with two range extenders. This may be down to varying levels of interference in the laboratory.

The transition from "reliable communication" or "flawless communication" to "poor communication" shows an improvement from the introduction of one range extender of around 20 dB (comparable to the PA gain of the range extender) and a further improvement from the introduction of the second range extender of around 10 dB (comparable to the LNA gain of the range extender. It is not clear why the arrangement with a single range extender maintains flawless communication at such high losses.

It would have been desirable to write specific test programs to allow longer tests with automatic recording of results and to build an experimental arrangement that can enclose all components in screened enclosures to eliminate the possible impact of external interference from other devices in the laboratory. This would allow gathering of more detailed information on how packet loss varies with signal loss.

7.5 Node enclosures

To allow deployment into the field the board containing the node electronics needed to be combined with a suitable battery pack and installed in a waterproof enclosure. The enclosure also provides probes on the bottom for measuring soil resistance. Contacts on the top are used for detecting water presence. A N-type connector on the top is used for connecting the antenna and a coax pigtail connects this to the main node hardware board.

The enclosures were designed in-house with the help of the School's Mechanical workshop. They are made from a section of PVC pipe. The pipe is intended to sit vertically in the ground. The bottom of the pipe is closed with a specially made plug which supports the screws that act as soil probes. This plug is solvent welded into the pipe to seal it before deployment. At the top of the pipe a flange is solvent welded on and the lid is attached to this flange. Screws and an o-ring are used to mount and seal the lid to the flange allowing it to be removed for maintenance. A desiccant sachet was placed inside the node enclosure to dry the air inside.

The bottom plug is hollow providing a space for wires leading to the bottom of the node enclosure. Inside the tube above the bottom plug is a machined PVC part that encloses the battery holder. Above this is the board containing the node electronics and finally there is a spacer piece to fill the remaining space in the node and accommodate the cable to the external antenna connector. The components of a node enclosure along with the node hardware PCB can be seen in Figure 7.8

A similar but larger enclosure was designed for the repeater node using a larger 110 mm pipe to accommodate larger batteries. This enclosure also contained a splitter from Mini-Circuits to allow two separate antennas to be connected. The repeater node enclosure did not include

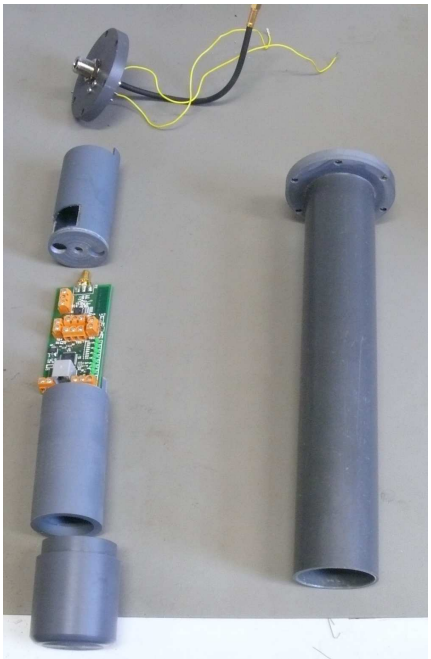


Figure 7.8: Components of the node enclosure

any sensing functionality.

Unfortunately the enclosures while fine during the short test deployment at the University proved problematic on the farm with water being discovered inside the sensor nodes. This will be discussed in further detail in Chapter 9.

7.6 Antennas

Initially the plan was to use sprung monopole antennas designed so they could survive being knocked over. The spring was made part of the radiating element. However these antennas proved to be difficult to actually make. Further it was decided that having the radiating element start from ground level was not sensible given the characteristic of Plane Earth propagation and the fact that the ground may not be perfectly flat. A photograph of one of these antennas can be seen in Figure 7.9.

For the deployments, monopoles formed from the end of lengths of coaxial cable were used. For the communication with end nodes the antennas were made from short lengths of coaxial cable and supported by thick heat shrink (it had initially been hoped that the stiffness of the cable alone would be sufficient but in practice it proved not to be). The base of the radiating element was approximately 150 mm from the top of the sensor node enclosures. A radiating element length of 4 cm (slightly longer than a quarter wave) was chosen based on tests with a vector network analyser. A photograph of one of these antennas can be seen in Figure 7.10. The inner insulation was left on the radiating element and heat shrink and silicone sealant were used in an attempt to waterproof the antenna.

The antennas for the link from repeater to base station were also monopoles formed from the end of coaxial cables and were mounted using GRP tubes. The radiating element of the antenna was approximately 1 m from the ground.

Figure 7.9: Sprung monopole antenna

Figure 7.10: Coax-end monopole mounted on buried node enclosure

7.7 Base station hardware

The base station is made up of a number of parts. The "node hardware board" as used in the sensor nodes provides the hardware and software to communicate with the sensor network. A BeagleBoard [16] running Linux, logs the data and runs the software needed to allow remote access to download the logs. Data storage for the BeagleBoard is provided by a SD card. At the time the BeagleBoard was one of the few affordable and easy to work with embedded Linux boards available. Since then newer boards have appeared that would probably be better suited for the application.

The BeagleBoard is attached to a BeagleBuddy Zippy Ethernet combo board [34]. The intention was to use the additional serial port, Ethernet port and battery backed real-time clock on this board. However the Ethernet port did not work (a hardware fault was suspected but it was never determined whether the fault was on the BeagleBoard or the Zippy) and the serial port turned out not to have any handshake lines. Therefore only the real time clock was used.

Access to the internet is provided by a Huawei E156G mobile broadband stick connected to an Edimax 3G-6200N mobile broadband router. This particular combination of broadband stick and router was chosen because it had been shown to work together by previous projects at the University and because it provided an external antenna connection. The BeagleBoard is connected to the mobile broadband router by a Farnell own-brand USB hub with Ethernet. Direct connection of the mobile broadband stick to the BeagleBoard should be possible but the software requirements of doing so were not investigated.

A custom board was designed and constructed to provide 12 V power for the mobile broadband router and 5 V power for the rest of the hardware in the base station. The power supply board also contains a USB to serial converter for communication between the BeagleBoard and the node hardware board. A handshake line on the USB to serial converter is connected to a transistor that allows the BeagleBoard to reset the node hardware board.

The power supply board went through two revisions. Initially it was believed that the base station would be run off a 24 V feed from a mains power supply. Based on this the Traco Power TSR 1-24120 and TSR 1-24120 DC-DC converters were chosen giving an input voltage range of 15 V to 36 V DC [117]. Later the decision was made to run off 12 V leisure batteries and to allow running off a single battery during battery changeover, it was decided to run the batteries in parallel (with diodes to prevent cross-charging). To accommodate this a second revision of the power supply board was designed with the TSR 1-24120 replaced by a Murata UEI15-120-Q12P-C allowing operation from 9 V to 36 V [76]. This converter also had an on/off control input, initially the control input was unused but later a transistor was added to allow turning the mobile broadband router on and off to extend battery life. This transistor was controlled by another handshake line on the USB to serial converter.

A picture of all the hardware in the base station can be seen in Figure 7.11.

Figure 7.11: Photo of basestation hardware

7.8 Software

The software on the nodes was based on the Microchip MiWi stack. The base station was made the "PAN coordinator", the repeater nodes[3] were made coordinators and the end nodes were made "full function end devices". It was decided to make the end nodes "full function" devices from the point of view of the MiWi stack even though they spent some time asleep since the "data request" system that MiWi uses for reduced function devices was considered unnecessary.

The nodes use the real time clock in the PIC to synchronise their activities. Setting the clock has a precision limitation of 1 second. The nodes wake up periodically and send a packet to the base station (possibly via a repeater), they then receive a response which they use to reset their clock. The precision limitation in setting the clock is seen as an advantage as it will prevent the nodes from waking up and attempting to transmit at exactly the same time leading to a packet collision.

The repeater also wakes up periodically but for a much longer time window to allow new nodes which do not have synchronised clocks to join the network. It would be possible to make most windows short and only have a long window once a day or so, this would reduce power use on the repeater nodes but would increase the time it took for a node that had lost synchronisation to rejoin the network and hence increase the data lost to network disruptions. The base station is always awake to avoid the complexities of shutting down and starting up the Linux system automatically.

The periodic wake-up is controlled by using the alarm function on the real time clock in chime mode. In this mode the real time clock provides a periodic alarm at intervals of every half-second, every second, every minute, every 10 minutes, every hour, every day,

[3]only one repeater node was actually deployed by there is nothing in the software design that requires that

every week, every month or every year. It was believed that once an hour was a reasonable interval for measurements in an agricultural deployment and as such this interval was used. During development shorter intervals of every 10 seconds or every minute were used so that the behaviour of the software could be tested more rapidly. The alarm setting is controlled by a constant stored in a file that is shared between the end node code and the repeater code so it can easily be changed in a single step.

When the end nodes and repeaters wake up they send a packet to the base station, in the case of the end nodes this packet contains sensor data. In the case of a repeater node the packet just notifies the base station of the wake-up. The base station replies with the current time thereby keeping the clocks roughly synchronised across the network. The clock in the PIC in the base station is synchronised to the clock on the BeagleBoard at network start-up. The BeagleBoard based system has a battery backed real time clock and was also set-up to use Network Time Protocol to synchronise its clock to real time.

End nodes are set to wake up when the significant fields of their RTC are zero (that is if set to hourly they wake up on the hour) and go to sleep either after receiving the time-sync packet or after a time-out. The time-out is set to 1 second but it could probably be shortened without any significant effects. Repeater nodes wake up for a 30-second window centred on the time when the nodes wake up.

After two failed communication attempts with the base station the MiWi stack is reset and the repeater or end node returns to searching for a network. When searching for a network nodes attempt to join the network every 25 seconds. Joining the network has a time-out of 100 ms which was determined based on looking at typical response times to network join requests and adding a safety margin. If the node can communicate directly with the base station it should join the network immediately. If the node cannot communicate directly with the base station it should join the network the next time the repeater node wakes up since the repeaters time window is longer than the nodes join attempt interval.

The PIC in the base station passes the data back to the BeagleBoard over a serial link. The BeagleBoard logs the Data in files on its SD card using a program written in pascal. The files can be retrieved from the BeagleBoard by using using SFTP over the mobile phone network. Since the mobile phone network does not supply a public IP address a program called vtund [55] is used to create a tunnel linking the BeagleBoard back to a private network at the University. Unfortunately this link proved to be somewhat unreliable in practice.

It was found during development that memory issues limited the size of the MiWi network table. Only 16 network table entries could be fitted in a data bank. Since the base station node will have every node on the network in its network table and a repeater will have all the nodes it serves in its network table the size of the overall network is limited by the network table size. The code was modified to allow the network table to be placed across two data banks and this could be extended further but memory for the network table is still likely to limit network size to around 150 nodes (the exact number has not been determined) with the current processor. If a revision to the design is made then a processor family with more memory should be considered to resolve this issue.

7.9 Power

The end node is equipped with a battery pack made up of a set of four AA cells connected in series. The repeater node is equipped with a battery pack made up of a total of 8 D cells. The D cells are grouped into two sets of four with the cells within a group connected in series and then the two groups connected in parallel. As discussed in Section 3.3.1, for a one and a half year deployment and assuming the batteries deliver half their nominal capacity this gives a maximum of 100 µA average current for the end node. Similar calculations for the repeater node yield a maximum acceptable average current of 1.36 mA.

Assuming measurements are transmitted to the network hourly and network searches are done every 25 seconds as described in the previous section the most power hungry state for an end node is when it is searching for a network since this is when it spends the most time awake. As a node may be cut off from the network for arbitrary lengths of time battery life calculations must assume that the node is always searching for a network. The most power hungry state for the repeater node is when it is waking up for 30 second intervals every hour.

To determine the power consumption of the end node in searching mode a reading 100 seconds long (4 search attempts) was taken using the power measurement system described in Chapter 5. The power consumption in network searching mode as measured by the power measurement system has a measured average of 190 µA (to two significant figures). This is about twice the 100 µA goal determined in the network design considerations chapter but is tolerable for test deployments. The network discovery time-out is set to 100 ms but from the power measurement data the node seems to be awake for around 150 ms. It is not presently clear why this is. When the node is awake the current sits at around 30 mA most of the time peaking at around 40 mA (presumably this peak corresponds to the transmission of data). Sleep current averages 26 µA. These readings were taken with a supply voltage of 5V.

For the repeater node direct measurements in the operating mode were not used due to the very long cycle time (1 hour) which would both make the measurements very time consuming and would likely overflow the limits of the measurement software (the raw data would be around 2GB and loading it into the current Matlab based processing software would result in a memory usage many times that). Instead values were calculated values based on the hardware's performance in the "searching for network state".

Without the range extender in place the node draws around 30 mA when awake. It will be awake $\frac{1}{120}$ of the time and asleep $\frac{119}{120}$ of the time and will be transmitting a negligible amount of the time, giving an average current of 276 µA. With the range extender in place and the power wasting pull down resistors on the range extender board removed, the sleep current is not noticeably changed by the range extender. The active current increased to about 32 mA and the transmit peak increased to about 81 mA, the actual peak current draw when running off batteries may be higher as it is believed that the power measurement system was limiting the delivered current during transmission. This increases the calculated average to 292 µA. These average figures are well within the power capabilities of the larger battery pack fitted in the repeater node. However the 81 mA peak is a cause for concern as it represents a brief overload of the 3.3 V regulator.

Since the repeater node is well within its power budget and the end nodes are exceeding their power budget during network search a possible fix would be to increase both the search

interval of nodes searching for network and the receive window of the repeater node. By doubling the length of the time window used by the repeater and also doubling the interval between connection attempts when a node is searching for a network, it is likely that the average current of the end node in searching mode could be reduced below 100 µA without violating the power budget of the repeater node.

7.10 Processing of data from nodes

The software running the base station saved incoming data into text files. To reduce the chance of data loss the data was stored in small files with each file covering one minute and the files named with date and time in the form YYMMDD_HHMM. Each line of the file contained data from the sensor node including the nodes ID, readings from the sensor and RSSI (received signal strength indication) and LQI (link quality indication) values from the RF transceiver in the base station. Nodes are identified by a 64-bit "long address" expressed in hexadecimal. In the deployment for this project addresses of the form DEADBEEF000000xx were used with xx being a sequential "node number" allocated to the node. To process this data into a more useful form for analysis a program was written in Java. This code is given in Appendix M

The only indication of message arrival time was the name of the files limiting resolution to one minute. Furthermore the timestamps in the files were rounded down to the next minute so messages arriving approximately on the hour (the time they were supposed to arrive) were split between two files, one covering the minute before the time point and one covering the minute after the time point. To group together the readings that arrived just before the expected time with the readings that arrived just after the expected time, the processing software was made to round the time to the nearest 10 minutes. This nicely grouped together those readings surrounding an expected time point for most points but there was a clock skew issue which is discussed in Section 9.3.4. As a result of this the time window was later widened to 30 minutes.

The processing has five stages. The first stage is to read the input data and determine what nodes have been seen. The list of nodes seen is used to determine the width of the table used for later processing and to provide a mapping between node "long addresses" and columns in the table.

The second stage is to read through the input data again and process it into a table of time points and readings. Each line in the table is represented as an array of type double while the table as a whole is represented as an ArrayList object. The first Five columns give the year, month, day, hour and minute of the data points. The sixth column gives the time since the start of the deployment in days. After that is a group of columns indicating the time in minutes since the last message was successfully received from each node (0 for nodes seen at the current time point, -1 for nodes that have not been seen at any previous time point) followed by groups of columns for the sensor values, first the two temperature sensors, then the soil resistance and water presence sensors and finally the LQI and RSSI readings.

During this stage readings from some sensors are also converted from raw values into meaningful units. Temperature readings are first converted from ADC units to millivolts based

on the assumption that the full scale voltage of the ADC is 3.3 V. The results are then converted to degrees centigrade by subtracting 400 mV and dividing by 19.5 mV/ deg C. This scale factor and offset was taken from the MCP9701A data sheet [70].

Soil resistance readings were converted from ADC units to resistances based on calibration factors gathered from a single node hardware board under conditions of short circuit and 10 kΩ resistance. Due to the design of the soil resistance circuit a 10 kΩ resistor should produce a half scale value at the divider. There is an offset to be expected due to volt drop in the diodes. To calculate the offset the reading with the 10 kΩ resistor was doubled and subtracted from the reading with a short circuit. This offset is then added to both the reading being processed and the open circuit calibration reading. The resistor divider equations are then used to calculate the soil resistance value.

RSSI readings were converted to dBm using a table created by linear interpolation of information given in the data sheet for the MRF24J40 [68]. The conversion processes for the temperature and RSSI measurements will be discussed in more detail when measurements using those sensors are presented in Sections 9.3.4 and 9.3.3. Readings from the water presence detection contacts and LQI readings are passed through in raw form as the author did not believe there was any meaningful conversion that could be performed on them.

The third stage is to perform interpolation to fill in missing values based on the previous and next known values thereby providing a value for every node at every time point. For time points before a node was first seen the first value from the node was used. For time points after the final time a node was seen the final value from that node was used. For other time points where a node was not seen linear interpolation was used between the previous and next time the node was seen.

The fourth stage is to write the data table created by the previous stages out to files. In the current version of the software two such output files are produced, one with all time points included and one with only time points where nodes 7, 8, 9, and 11 reported successfully. The reason for focusing on these nodes will be discussed later when the temperature measurements from the deployment are discussed.

The final stage is to compute message loss statistics. These are generated based on 5 day periods and will be discussed in more detail in the section describing the message loss results.

Chapter 8

Preparation for fieldwork

The main goal of the project was to investigate the operation of wireless sensor networks in a rapeseed oil crop. This was to be achieved by making measurements of the propagation loss through the crop and by performing an experimental deployment of a sensor network within the crop. In preparation for these activities, preliminary outdoor measurements and equipment testing was needed.

The goal of these preparatory activities was to gain experience with making measurements and operating sensor node equipment outside the laboratory and determine approximately what distance signals could be transmitted through the rapeseed crop using the hardware described in Chapter 7. An idea of what distance signals could be transmitted in the crop was needed to plan later deployment and measurement activities.

It had been hoped to complete the preparatory work and perform an actual deployment sufficiently early to allow for two deployments in a rapeseed crop. The intention was to have the first deployment in the second year (2010-2011) of the PhD and another in the third year (2011-2012). In this way lessons learnt from the first deployment could be applied to the second deployment. Unfortunately organisational difficulties and delays meant that only one deployment was possible.

In April 2011 a three day field trip was made to Rothamsted Experimental Station [95]. This trip served two purposes. The first was to obtain experience operating the propagation measurement system described in Chapter 6 in the field. The second was to make initial measurements that would give some indication of the distance radio signals could be transmitted within a rapeseed crop. Originally the plan had been to perform the deployments and final propagation measurement experiments there as well but the long distance between Rothamsted experimental station and the University made this impractical to organise.

Later arrangements were made to work with a local farmer, Richard Reeves who runs Tatton Dale Farm and plans were made to perform a deployment there. To test the nodes and associated base station and repeater hardware outdoors before the deployment at Tatton Dale Farm, there was a deployment at the University in February 2012.

8.1 April 2011 propagation measurements

The initial field trip was intended to serve two purposes, the first was to become familiar with using the propagation measurement equipment described in Chapter 6 in a field situation, the second was to gain sufficient measurements of propagation loss in rapeseed to plan deployment of an experimental network with distances that would give useful results. If the distance between nodes in a deployment was too large then no signal would get through at all giving no results. If the distance between the nodes was too small then the data may not show any effects from the propagation loss in the crop. Both cases would provide less informative results than a system that was operating at distances that were close to the limit of what the RF transceiver hardware in the nodes could tolerate.

The work, including travel to and from Rothamsted Experimental Station [95] happened over the course of three days starting on Tuesday 26$^{\text{th}}$ April 2011 and finishing on Thursday 28$^{\text{th}}$ April 2011. On Tuesday initial experiments and equipment tests were performed in a rapeseed field. The field propagation measurement system was operated as part of this initial testing but distances between antennas were not measured and results were not recorded.

The initial testing raised some concerns. The first was related to measuring distance. The initial plan had been to measure distance using a Leica Disto A2 laser distance measure. However this proved to be unusable in the sunny weather leaving no available method to measure distances. The other was that measurements of the signal propagating between the two antennas even without the crop between the antennas were significantly lower than

Figure 8.1: Field of rapeseed in which experiments were performed.

the author had (naively) expected based on the free-space propagation model. The author was vaguely aware of the Plane Earth propagation model at this time but its implications and the fact it was more appropriate than free space propagation had not yet been fully appreciated.

The antennas used during the initial tests were the "sprung monopoles" described in Section 7.6. These were used because at the time it was planned to use them on the sensor nodes. These antennas were a fairly unusual design in that while the spring and wire making up the monopole were electrically connected, the connection was only at the bottom and as such it was suspected they may be the cause of the worse than expected propagation measurements.

After these experiences several decisions were made. Firstly a surveyors tape measure was purchased so that distance measurements could be made. Secondly the antennas were replaced with ones of a more conventional design. These were constructed from from coax intended for satellite TV with the end stripped to produce a monopole. This was considered less than ideal because the impedance of the satellite TV coax was 75 ohm which is not matched to that of standard RF test equipment but it was the best that could be constructed given the limited available resources. Thirdly it was decided to perform tests in an open field to find out the loss that could be expected without the crop.

Wednesday was spent making the measurements in an open grass field. Thursday was spent taking measurements in the crop both near the ground and at a higher level approximately level with the top of the rapeseed canopy. Details of these activities will be given in Sections 8.1.1 to 8.1.3.

During and prior to the trip the weather was very dry and sunny. The rapeseed crop was in full flower with the majority of plants being about 90 cm tall and the tallest plants being about 110 cm tall. A picture of the field can be seen in Figure 8.1.

8.1.1 Open field measurements

The 433 MHz, 868 MHz and 2.4 GHz ISM bands were tested with a pair of coax monopole antennas constructed from from coax intended for satellite TV with the end stripped to produce a monopole. The length of the wire was cut to be a known fraction of a wavelength for each test and was gradually cut down as the tests progressed for higher frequencies and smaller fractions of a wavelength such that by the end of the day it was cut to be a quarter wavelength at 2.4 GHz. All tests were done with a horizontal distance of 10m between the antennas and with the base of the radiating element 10.2 cm off the ground.

In retrospect this was a poor choice of parameters. Limited field time meant that frequency bands other than 2.4 GHz were never tested in the crop. Furthermore 10 m was much longer than the distances found to be feasible for in-crop transmission, so the results from these tests could not be directly compared with any of the author's later tests within the crop.

The improvised coax monopole antennas were mounted on the surface stands as shown in Figure 8.2 which were placed on the ground. The first tests were performed both with and without a ground plane made up of braid stripped from the coax. The ground plane assembly consisted of 8 radials each 50 cm long. However the presence or absence of the

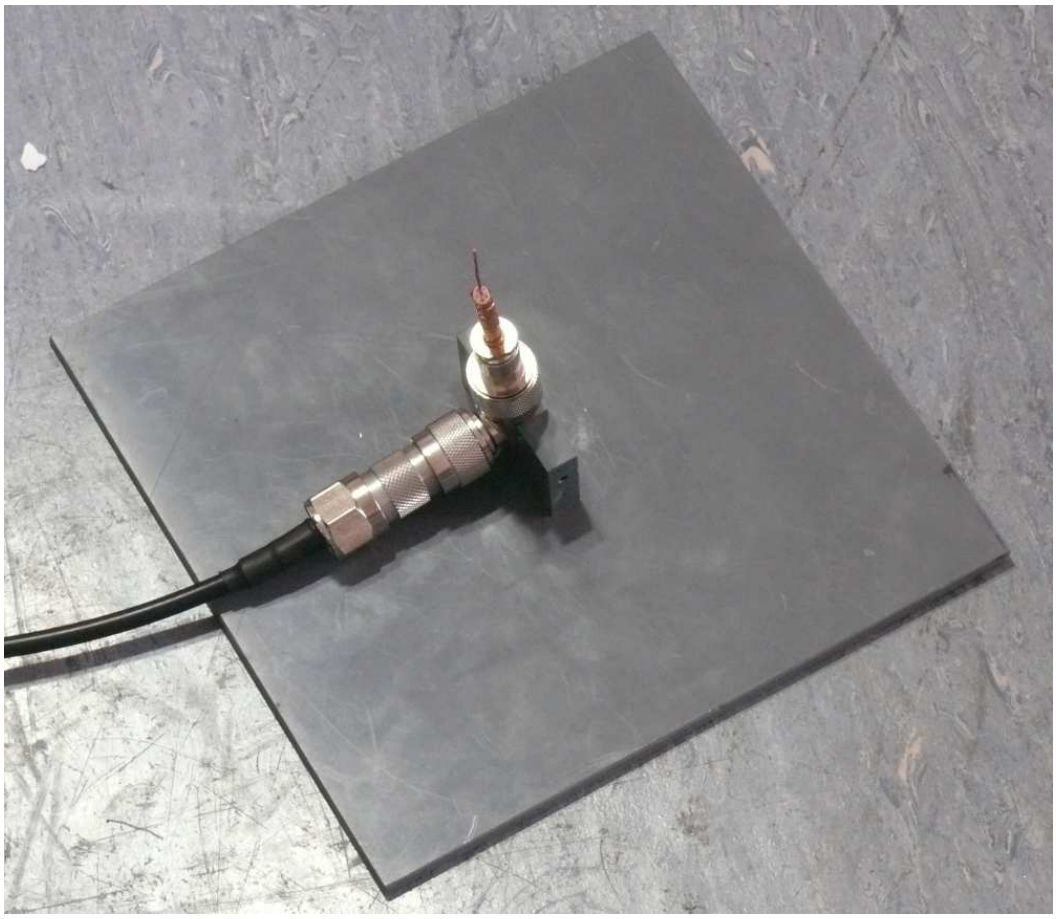


Figure 8.2: Coax monopole antenna on surface stand

ground plane assembly had no noticeable impact on the propagation, so further tests during the measurement session were only performed once (with the ground plane in place). The ground plane assembly was not used at all during later measurement sessions.

Measurements can be seen in Table 8.1 along with values of path loss calculated using the Plane Earth propagation model described previously in Section 3.5 and assuming no antenna gain [1]. The author selected the Plane Earth model as a well known model that was appropriate to situations with a ground surface present. Actual measurements showed between 10 dB and 30 dB more loss than the Plane Earth calculated values. It is clear that having the antennas at such a low height leads to very poor propagation even in an open grass field. It was therefore decided to increase the antenna height for the deployment by using monopoles formed from the end of a section of coax rather than monopoles where the radiating element starts at the connector.

8.1.2 Measurements through crop near ground level

The first measurements made on Thursday were through the crop near ground level using the same surface stands and improvised coax monopole antennas used in the grass field

[1] crude antennas like the ones used here can be expected to have very low gain or even possibly loss

nominal frequency (MHz)	height of base of radiating element (m)	wavelength (m)	wavelength fraction	height of center of radiating element (m)	separation (m)	Plane Earth path loss at 10m (dB)	measured minimum propagation (dB)	measured maximum propagation (dB)	measured average propagation (dB)
433	0.102	0.693	0.375	0.362	10	57.7	-67.9	-67.3	-67.5
868	0.102	0.346	0.625	0.318	10	59.9	-78.6	-71.9	-74.3
868	0.102	0.346	0.5	0.275	10	62.4	-79.8	-76.9	-77.9
868	0.102	0.346	0.375	0.232	10	65.4	-67.9	-67.3	-67.5
2400	0.102	0.125	0.625	0.180	10	69.8	-96.0	-81.5	-87.1
2400	0.102	0.125	0.5	0.165	10	71.4	-91.7	-81.8	-86.6
2400	0.102	0.125	0.375	0.149	10	73.1	-92.6	-85.6	-90.2
2400	0.102	0.125	0.25	0.133	10	75.0	-105.9	-85.2	-94.6

Table 8.1: Summary of grass field measurement data including Plane Earth figures for comparison

measurements but without the ground plane assembly. After the previous activities in the open field the antennas were a quarter wavelength at 2.4 GHz and they were kept at this length. This placed the centre of the radiating element approximately 13 cm off the ground. All measurements were made in Furze field. A map showing the location of this field can be found on page 26 of the Rothamsted publication "Guide to the Classical and other Long-term Experiments, Datasets and Sample Archive." [94].

Measurements were only performed in the 2.4 GHz band due to time constraints and the fact that 2.4 GHz was already planned for the deployment. Measurements were performed with distances between the antennas of 1 m, 2 m, 3 m, 4 m and 5 m. The 1 m and 2 m tests were performed with a frequency step of 10 MHz, while the remaining tests were performed with a narrower step of 1MHz to investigate fading effects. It would have been desirable to have performed all the tests with the narrower frequency step but time did not permit that. The frequency range was 2.4 GHz to 2.5 GHz.

The results for these experiments can be seen in Figure 8.3. Note that in the original dataset there was a failed measurement at 2.458 GHz in the 4 m dataset. Therefore this data point was interpolated for the purpose of plotting the graph. The results of the 2 m test do not seem to fit the trend formed by the other tests and for many frequencies are lower than the 3 m test. A possible explanation could be that the 2 m test happened to be made through a particularly dense section of foliage.

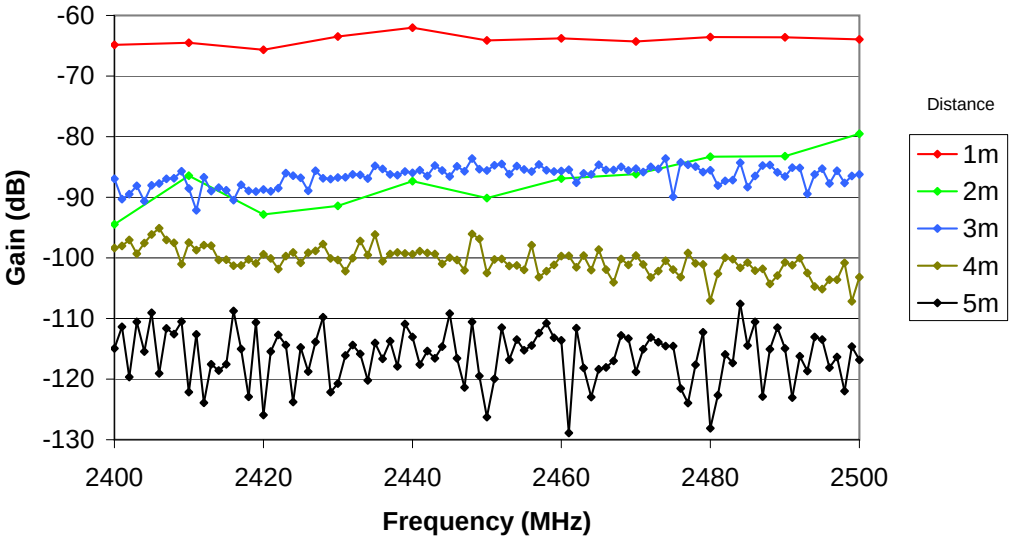


Figure 8.3: Propagation measurements through crop base

These measurements made it very clear that transmission through the crop near the ground at 2.4 GHz was only viable for short distances. The losses in the 4 m tests being approximately on the limit of what typical wireless sensor hardware can operate with even with the use of two range extenders[2]. As well as an increase in loss, there is an increase in fading with

[2]The author believes that the best way to determine how much loss a given RF system can operate acceptably with is to test it directly by increasing the attenuation between two transceivers until the system fails. See Section 7.4 for link budget measurements

distance with the 5 m test showing especially bad fading of around 20 dB. This suggests that spread spectrum modulation techniques should be used for in-crop communication systems.

8.1.3 Measurements through crop canopy

After discovering that only very short transmission distances were likely to be possible through the crop near the ground a new approach was considered. That of placing the antennas at a height which was above the top of the majority of the plants in the crop but below the top of the tallest plants. The author refers to this as transmitting at "canopy level". Due to the fact this had not been considered in advance the antenna stands had to be improvised. Figure 8.4 shows three side views of the crop with a tape measure taken at different heights. These show that the bulk of the plants were approximately 80 cm to 90 cm high but there were occasional plants that were taller, sometimes as high as 110 cm.

The measurements through the crop at canopy level were taken with a transmit antenna height of 97 cm and a receive antenna height of 93 cm on improvised stands made from sticks that were found near the field. The same antennas were used as in the measurements near ground level. Measurements were taken at separation distances of 10 m, 20 m and 30 m with 11 points in each measurement set. The frequency range was 2.4 GHz to 2.5 GHz. A picture of the antenna setup used can be seen in Figure 8.5. Results are shown in Figure 8.6.

These results show that by moving the antennas up into the crop canopy far greater range can be achieved than with the antennas near the ground. The results show about 10 dB of frequency selectivity. Based on these results and the link budget measurements discussed in Section 7.4, it should be possible with the use of range extenders to achieve wireless sensor network operation at 10 m, under the conditions[3] used in this experiment with simple antennas.

Figure 8.4: Side views of rapeseed crop with tape measure.

[3] including the state of the crop

Figure 8.5: Antenna arrangement used for measurements through crop canopy

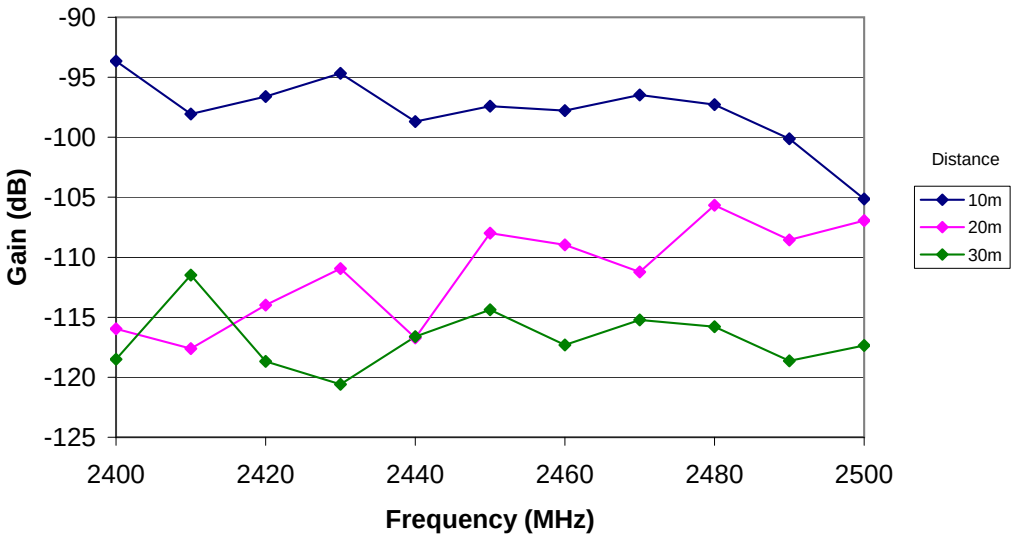


Figure 8.6: Measurements through crop canopy

8.1.4 Comparisons with two-ray model

To provide a point of comparison and to potentially address the anomalous results from the 2 m test near ground level, the measurements from the crop at Rothamstead were compared with a two-ray model. This model provides some indication of the loss expected without the crop present and hence provides some indication of how much additional loss was being caused by the crop.[4] The two-ray model assumes that the signal at the receiver is formed by combining signals received over two paths, one path direct from transmit antenna to receive antenna and one path reflected by a perfectly flat ground surface. This model is commonly used as a simple approximation for propagation loss when transmitting over the ground. An equation for the two ray model is given in Equation 8.1 [99].

$$|p(t)| = \sqrt{G_t G_r}\frac{\lambda}{4\pi}\left|\frac{e^{-j2\pi l/\lambda}}{l} + \Gamma\frac{e^{-j2\pi l'/\lambda}}{l'}\right| \tag{8.1}$$

$p(t)$ is the ratio of the received electric field divided by the transmitted electric field. $|p(t)|^2$ is equal to the receive power divided by the transmitted power. l is the length of the straight line path between the two antennas. l' is the length of the reflected path between the two antennas. Γ is the reflection coefficient (which will be taken as -1 assuming perfect reflections). λ is the wavelength. G_t and G_r are the transmit and receive antenna gains and will be taken as 1 assuming that the simple monopole antennas used in the field work had negligible gain.

Taking into account the assumptions and substitutions mentioned above the equation simplifies to the form given in Equation 8.2

[4] Note that these calculations were undertaken after all field work had been performed and so did not serve to influence later field activities.

$$P_r/P_t = \left(\frac{\lambda}{4\pi} \left| \frac{e^{-j2\pi l/\lambda}}{l} - \frac{e^{-j2\pi l'/\lambda}}{l'} \right| \right)^2 \tag{8.2}$$

And converting the result to decibel form (that is redefining P_r and P_t as being in decibel units rather than linear units) gives Equation 8.3

$$P_r - P_t = 10 \log \left(\left(\frac{\lambda}{4\pi} \left| \frac{e^{-j2\pi l/\lambda}}{l} - \frac{e^{-j2\pi l'_0/\lambda}}{l'} \right| \right)^2 \right) \tag{8.3}$$

The length of the direct and reflected paths can be calculated from the two heights (h_1 and h_2) and horizontal separation (d) of the antennas as shown in Equations 8.4 and 8.5.

$$l = \sqrt{d^2 + (h_1 - h_2)^2} \tag{8.4}$$

$$l' = \sqrt{d^2 + (h_1 + h_2)^2} \tag{8.5}$$

$P_r - P_t$ represents the "Gain" of the channel. Since the channel is a passive structure the "Gain" figure should always be negative indicating a loss.

This model is similar to the plane earth model mentioned previously in Section 8.1.1 but unlike that model which only considers the impact of destructive interference at long distances the two ray model given above considers both constructive and destructive interference at shorter distances potentially resulting in a pattern of peaks (where the signal constructively interferes) and nulls (where it destructively interferes) with distance. In particular it was suggested that this could possibly explain the extra signal loss in the 2 m measurements near ground level and/or the substantial frequency selective fading in the author's results.

For the experiments near ground level, the antenna height was taken as 0.133 m based on measurements of the antenna/stand combination used in the tests. A plot of loss versus distance was made for these conditions at 2.45 GHz and is shown in Figure 8.7. Only a single null is seen on the graph at approximately 0.24 m followed by a peak at approximately 0.5 m then a consistent decline in signal level up to 5 m. Since the observed null and peak are at a substantially shorter distance than any of the author's measurements this model does not explain the anomalous results of the 2 m test near ground level.

The losses predicted by the two ray model for the antenna height of 0.133 m, frequencies of 2.4 GHz, 2.45 GHz and 2.5 GHz and the distances used in the author's measurements through the crop are shown in Table 8.2. While the model does have some frequency dependence the loss predicted by the model at the distances corresponding to the author's measurements did not change substantially over the frequency range of 2.4 GHz to 2.5 GHz. Therefore this model does not explain the frequency selective fading seen in the author's measurements. A comparison of the model results with the measurements from the crop is shown in Figure 8.8. As expected with the crop present there is substantial additional signal loss over that predicted by the 2-ray model, especially at larger distances. The 2 m results still do not fit with the trend of increasing signal loss with increasing distance.

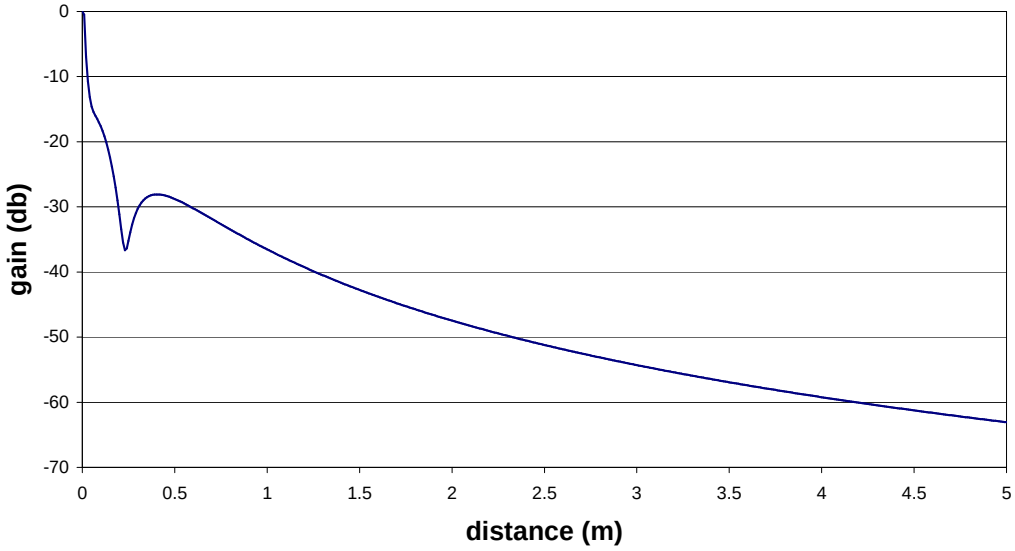


Figure 8.7: Channel gain predicted by two-ray model verses distance for antenna height of 0.133 m and frequency of 2.45 GHz.

		Frequency		
		2.4 GHz	2.45 GHz	2.5 GHz
Distance	1 m	-36.5 dB	-36.5 dB	-36.6 dB
	2 m	-47.4 dB	-47.5 dB	-47.5 dB
	3 m	-54.3 dB	-54.3 dB	-54.3 dB
	4 m	-59.2 dB	-59.2 dB	-59.2 dB
	5 m	-63.1 dB	-63.1 dB	-63.1 dB

Table 8.2: Channel gain predicted by two-ray model for frequency, height and distance used in measurements at low level through the crop.

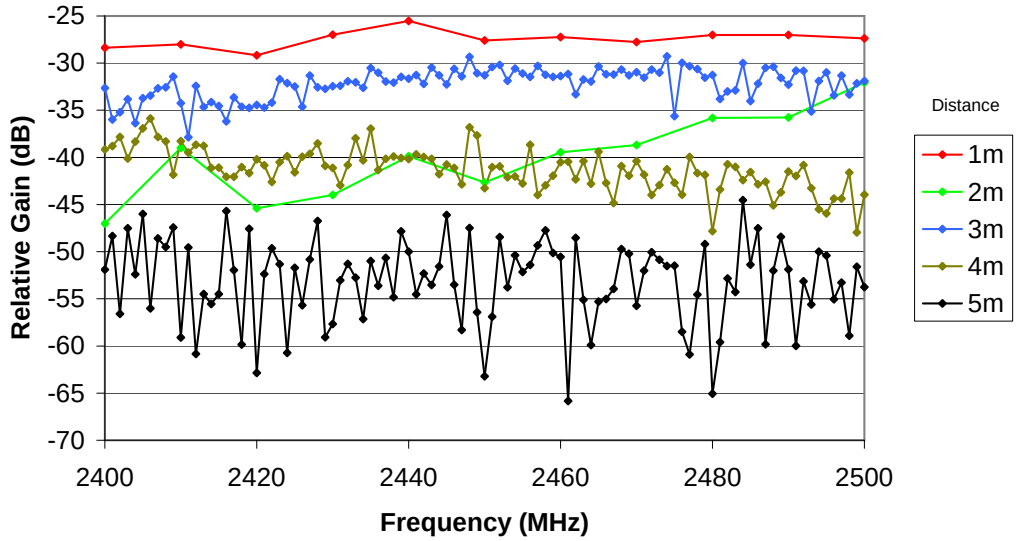


Figure 8.8: Difference between channel gain predicted by two-ray model and channel gain measured in the crop near ground level. Negative values indicate that the loss observed in the crop is greater than the loss predicted by the two-ray model.

For the experiments near crop canopy level, the transmit antenna height was recorded as 0.97 m and the receive antenna height was recorded as 0.93 m. The results of inserting these values along with a frequency of 2.45 GHz into the model are plotted in Figure 8.9. The greater antenna height (and associated greater path length difference) results in a much larger number of peaks and nulls than in the near-ground scenario. However while there are a large number of nulls on the graph, the distances at which the author took measurements do not coincide with any of them.

The losses predicted by the model for a transmit antenna height of 0.97 m and a receive antenna height of 0.93 m, the distances used in the crop canopy experiments and frequencies of 2.4 GHz, 2.45 GHz and 2.5 GHz are given in Table 8.3. The model shows more variation with frequency in this case than it did for the near-ground scenario, but the variation with frequency predicted by the two-ray model is still very small compared to the variation with frequency observed in the experimental data. A comparison of the model results with the measurements from the crop is shown in Figure 8.10.

		Frequency		
		2.4 GHz	2.45 GHz	2.5 GHz
Distance	10 m	-54.3 dB	-54.3 dB	-54.5 dB
	20 m	-62.3 dB	-62.9 dB	-63.4 dB
	30 m	-63.6 dB	-63.8 dB	-63.9 dB

Table 8.3: Channel gain predicted by two-ray model for frequency, height and distance used in measurements at canopy level through the crop.

While the two-ray model only provides a crude approximation of the loss that would be expected without the crop present, the comparisions for both the measurements near ground

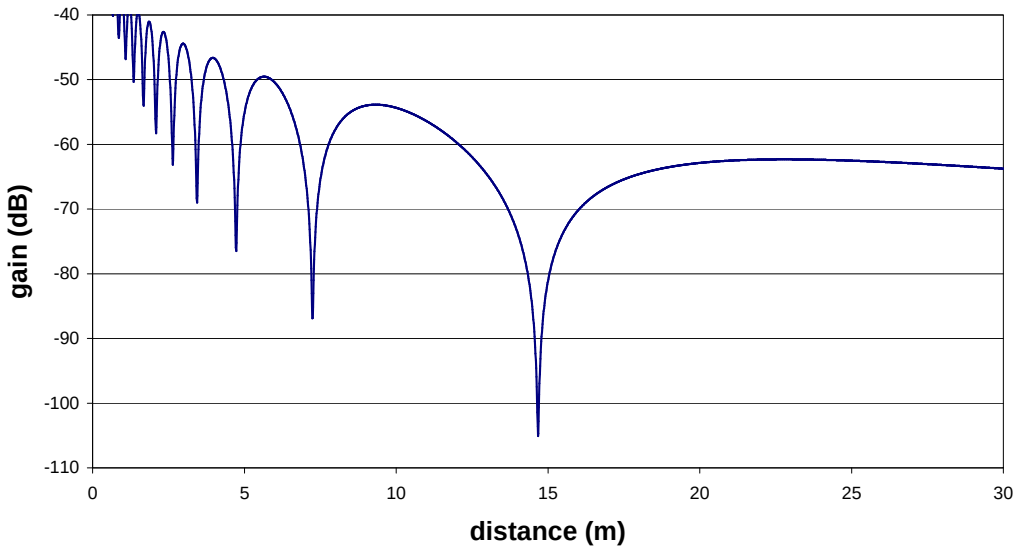


Figure 8.9: Channel gain predicted by two-ray model verses distance for transmit antenna height of 0.97 m, receive antenna height of 0.93 m and frequency of 2.45 GHz.

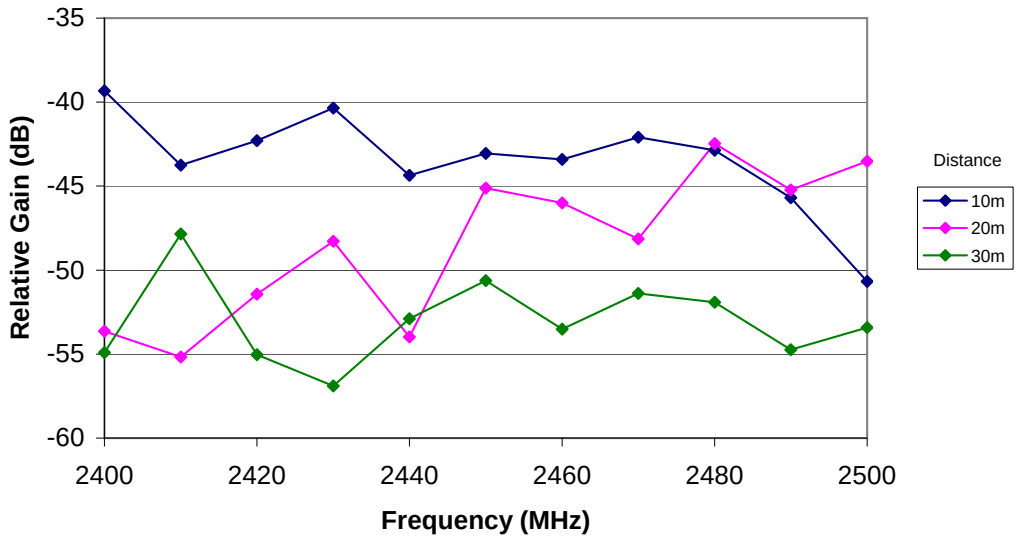


Figure 8.10: Difference between channel gain predicted by two-ray model and channel gain measured in the crop canopy. Negative values indicate that the loss observed in the crop is greater than the loss predicted by the two-ray model.

level and the measurements in the crop canopy indicate that substantial additional loss is introduced by the crop. Furthermore the substantial frequency selective fading cannot be explained by the interference of a direct ray with a single ground reflection and therefore indicates that there are almost certainly more than two paths involved in the signal propagation through the crop.

8.2 Deployment at the University

On 29 February 2012, a wireless sensor network was deployed into a non-public garden area behind the Faraday building at the University of Manchester, for equipment testing purposes. It had been hoped to do this much earlier but it took a long time to get permission from the Estates Department at the University. The purpose of this deployment was to gain experience operating the equipment outdoors, identify potential problems and decide whether it would be feasible to install the nodes at the farm in single session. While the garden area was not public and access was supposedly controlled by a locked gate the site was not securely fenced on all sides.

It had been hoped to arrange the equipment so that the end nodes had a line of site path to the repeater but not to the base station. However physical constraints on where the network could be deployed prevented this. Specifically the Estates Department at the University wanted the high value equipment to be in a more secure yard that was separated from the garden by a locked gate and we could only make holes for the nodes in bare soil areas not on the grassed lawns. The deployment consisted of the base station with high level cellular and wireless sensor network antennas[5], the repeater node with high and low level antennas and four end nodes with low level antennas.

The antennas used were formed from the end of coaxial cables as described previously in Section 7.6.

The base station hardware was located in the yard area behind a locked gate. The two antennas for the base station were located in the garden area next to the gate. The repeater node was located at the edge of a bare soil area near the gate and the four end nodes were then located in a line from the base station. A photograph of the deployment showing the antennas for the base station, the repeater node with high and low level antennas and three of the end nodes is shown in Figure 8.11. An extract from the University campus map showing the approximate location of the Deployment is shown in Figure 8.12.

To increase the number of data points gathered over the short period available the nodes were programmed to wake-up every 10 minutes. This is 6 times more frequent than the once per hour wake-up frequency that was used for the deployment on the Farm.

The data gathered by the base station was processed to produce a table of time points (times rounded to the nearest 10 minutes) and which nodes had responded at those time points. However there was not seen to be any need to calculate statistics. Transmission was reliable with only occasional lost messages (the exact number was not counted as it was not deemed to be of interest). The reliable transmission was expected as there were no obstructions between the antennas. There were however a couple of problems during the deployment. Firstly Node 1 failed about a day into the deployment. It is believed this was due to an internal wiring error that caused the node to draw far more battery current than it should.

It also appears the base station ran out of battery power after just under a week (the first reading from the deployment at the University was on the 29th of February 2012 at 12:39, the last was on the 7th of March 2012 at 10:50). This is less than the 10 days anticipated

[5] a terminator was fitted on the connector that would normally hold a low level wireless sensor network antenna on the base station

Figure 8.11: Photograph of deployment at the University.

Figure 8.12: Approximate location of deployment at the University. Building 19 is the Faraday Building.

from battery discharge measurements and was concerning given that we had agreed with the farmer to have one battery swap per week. It is not clear what the cause of this was. Poor mobile reception, causing the mobile broadband equipment to consume more power than expected is suspected. It is also thought that credit on the mobile broadband equipment may have run-out part way through the deployment causing power to be wasted on futile reconnection attempts. Another possibility is that the leisure batteries used to power the base station had poorer performance in the cold outdoor conditions than when tested in the laboratory.

Chapter 9

Tatton Dale Farm deployment and measurements

As stated earlier the main goal of the project was to investigate the operation of wireless sensor networks in a rapeseed oil crop. In particular the aim was to determine what transmission distances were feasible with simple antennas located close to the ground for a network that is expected to operate throughout the growing season. Transmission with the antennas placed further above the ground was also investigated, but in less detail.

Some measurements of signal loss in a rapeseed oil crop had already been gathered during the trip to Rothamsted in 2011. These measurements were made on a single day and therefore it was not known if they represented the time when signal loss due to foliage was at its greatest. Furthermore the complete behaviour of a radio channel is more complex than can be expressed by an attenuation figure alone. As such while combining channel loss measurements with measurements of how much signal loss wireless transceivers can operate reliably with (See Section 7.4) provided a first approximation of feasible distances, it was decided to validate this with actual operation of a wireless sensor network.

The plan was to take two approaches. Firstly measurements would be made with the propagation measurement system described in Chapter 6 on multiple occasions. These measurements would provide direct measurements of the channel loss and give indication of the level of variation with small changes in frequency, variation between different areas of the crop and variation between different dates but the number of dates would be limited by the practicalities of arranging trips to the farm.

Secondly, an experimental deployment of a wireless sensor network was to be performed within the crop. This would provide direct information on how the message loss rates varied with time and distance and would also provide crude measurements of received signal strength at the base station. Provided the hardware functioned correctly it would provide this information continuously throughout most of the growing season. It would also allow the author to gain experience on the practicalities of operating a wireless sensor network outdoors. While the nodes were fitted with sensors, collection of sensor data was considered a side activity, not the main purpose of the deployment.

The deployment and measurements were performed at Tatton Dale Farm from March to August 2012 with measurements outside the crop for comparison purposes gathered in 2013. Activities took place on the edge of a field next to a farm track. The location was at a latitude of 53.3287° North and a longitude of 2.3544° West.

The initial deployment of sensor nodes was made on the 15th of March 2012 and the intent was to return approximately every 2 weeks to fix any problems that developed with the network and make propagation measurements using the measurement system described in chapter 6. Unfortunately due to weather and bureaucratic issues the actual visits were both far less frequent and not regularly spaced. In particular between the initial deployment and the second visit the crop had gone from being scattered small plants to being near fully grown. As a result of this, the opportunity to make propagation measurements during the most interesting part of the season was effectively missed.

The antennas used were formed by stripping the sheath and braid from the end of coaxial cables leaving the centre core to form a monopole and using heatshrink to seal and support the assembly as described previously in Section 7.6. These antennas were used for both the deployment and for the propagation measurements. For propagation measurements near the ground the antennas were mounted on the ground stands while for propagation measurements at higher level the telescopic stands were used. These stands were described in Section 6.3.

The addition of additional nodes to the network with their software modified to allow measurement of received signal strengths at points other than the base station was considered. This idea was abandoned in light of the difficulty arranging visits to the Farm, the crop damage that would be caused by adding nodes late in the season and the need to troubleshoot problems that had arisen with the nodes already deployed.

9.1 Activity log

9.1.1 First visit - 15th March 2012

On 15th of March 2012 a network was deployed at Tatton Dale Farm. Included in the deployment were a base station, a repeater node and a total of 12 end nodes. The end nodes were located in two lines one extending outwards from the base station and the other extending from the repeater node.

The deployment plan was as follows; the base station was to be located at the edge of the field and the repeater node was to be located 10 m from it (formed from 8 m along the edge of the crop and 6m further into the crop). There were to be two lines of end nodes, one extending from the base station into the field and the other extending from the repeater parallel to the edge of the field and away from the base station. Each line of nodes was to contain 6 nodes and the distances from the base station/repeater to the end nodes were to be 1 m, 1.5 m, 2 m, 2.5 m, 3 m and 3.5 m. Nodes 1-6 were in the line from the repeater while nodes 7-12 were in the line from the base station. Within each line the lowest numbered node was closest to the base station and the highest numbered node was furthest from it. Figure 9.1 has a diagram of this arrangement.

The deployment went far quicker than expected due to the soil at the farm being far softer than the soil in the area used for the deployment at the University. Holes were made for the nodes using a manual auger and then the nodes were inserted in the ground. Unfortunately the shape of the holes meant it was not practical to control the soil packing at the bottom of the holes leaving highly variable conditions for the soil resistance probes.

Due to a combination of measurement inaccuracies, finding suitable places among the plants to auger in the nodes and the finite size of the base station and repeater equipment the idealised measurements mentioned above could not be exactly followed. Generally measurements were made from the points on the devices considered important for that measurement. For example the distance from the end nodes to the repeater/base station was measured from the antenna on the node to the low level antenna on the base station or repeater. The distance between base station and receiver was measured between their high level antennas.

To document the difference between the idealised deployment plan and what was actually deployed distances were measured after all hardware was installed. These measurements are shown in Figure 9.2.

A photograph of the entire deployment is shown in Figure 9.3. The base station and large antennas are clearly visible in this photograph and the antennas for the end nodes near the base station are visible but are difficult to see due to their small size and the camouflage provided by the crop. A photograph of the repeater node and associated end nodes is shown in Figure 9.4.

After deployment most of the the nodes were found to be operating as expected. However two nodes (numbers 10 and 12) did not appear in the data collected by the base station at all. The decision was taken to remove these nodes and replace them on the next visit.

9.1.2 Problems with the network between first and second visits

While it had been hoped to visit the deployment once per fortnight a number of organisational issues prevented this and there was an undesirably long period between the first visit and the second. During this period a couple of problems developed with the network.

The nodes placed with the repeater largely stopped appearing in the data gathered by the base station (some sporadic readings were seen afterwards but nothing consistent) after the 4am reading on 11 April 2012. It was suspected at the time that the high level antennas had been covered by the foliage and plans were made to raise their height to 1.5 m at the next visit to the Farm.

During the week commencing 16th April 2012 all attempts to contact the network and download data failed. The previous successful download having been on Thursday 12th April 2012. It was not clear why but it was suspected that the farmer may have forgotten to charge the batteries (the farmer had said he would switch out the batteries on Mondays), connection to the network was re-established on the 23rd April 2012. When the data was downloaded a four day gap in the data was seen. The gap ran from Saturday 14th April 2012 to Wednesday 18th April 2012 with the first log file after the gap indicating that the network had restarted.

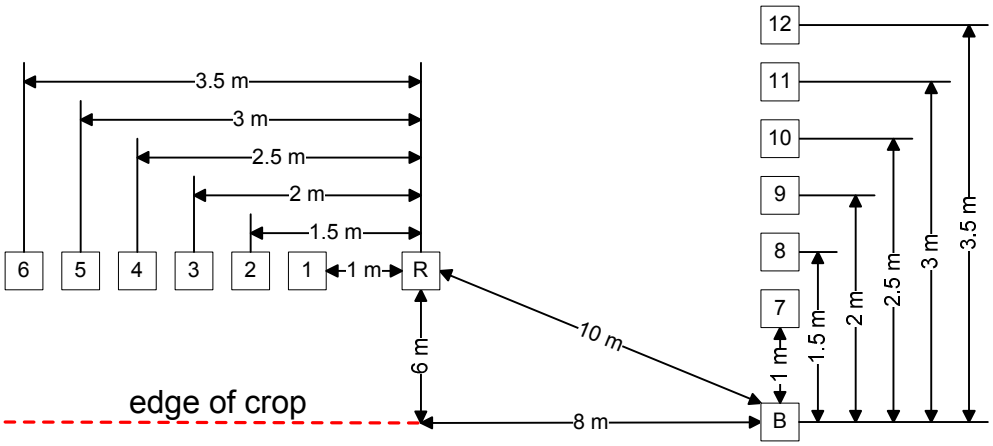


Figure 9.1: Diagram of intended layout of equipment in Tatton Dale Farm deployment, B is the base station, R is the repeater and numbered boxes are end nodes.

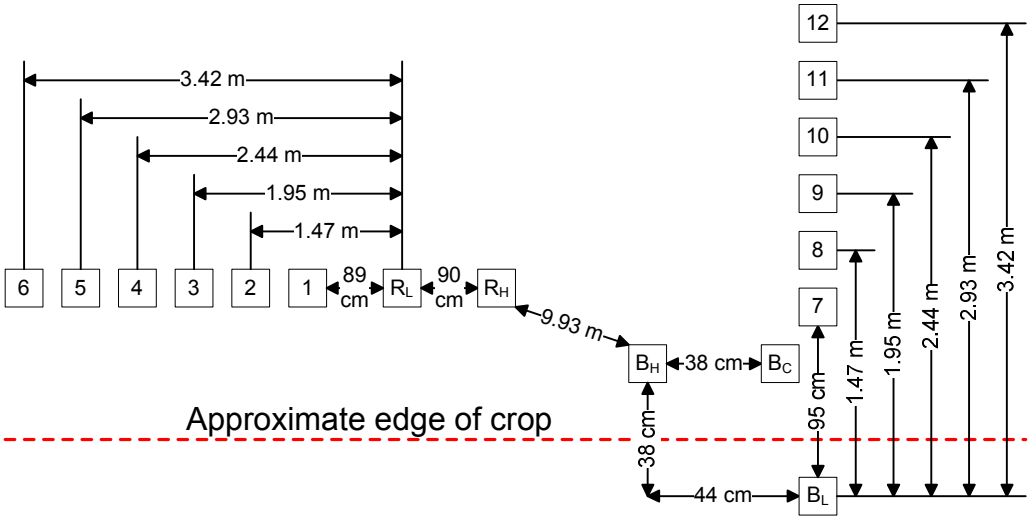


Figure 9.2: Diagram of actual layout of equipment in Tatton Dale Farm deployment, B_L is the low level antenna of the base station. B_H is the high level wireless sensor network antenna of the base station. B_C is the cellular antenna of the base station, R_L is the low level antenna of the repeater, R_H is the high level antenna of the repeater and numbered boxes are end nodes.

Figure 9.3: Photograph of entire farm deployment taken from near base station. The base station equipment is visible in the bottom left of the picture. The antennas for the end nodes near the base station are just about visible in a line from the base station. The high level antenna for the repeater can be seen on the right. The end nodes near the repeater are too small to see.

Figure 9.4: Photograph of repeater and associated nodes

9.1.3 Second visit - 3rd May 2012

On the 3rd of May 2012 a second visit was made. By this visit the rapeseed crop was in full flower. During this visit the two failed nodes were replaced and brought back to the University. Node 10 was replaced by node 13 and node 12 was replaced by node 14.

It was discovered that the high level antennas were still above the crop but that the crop was getting close to potentially covering them. As such it was decided to raise them to 1.5 m as planned. A photograph of the crop with the newly raised high level antennas is shown in Figure 9.5. It was decided that the next step would be to remove the repeater node from the field and check the batteries. However the tools for doing so had not been brought so this could not be done until the next visit.

Figure 9.5: Photograph of crop and high level antennas during second visit to farm.

During the trip it was discovered that there was no power to the base station. This was reported to the farmer a few days later and after the network recovered the data was downloaded and as expected showed a gap in the readings. The last reading before the gap was at 00:00 on the 1st of May while the first reading after the gap was at 08:00 on the 10th of May. When asked about this the farmer claimed to be changing the base station batteries regularly.

After the network came back up the two new nodes appeared but the repeater did not reappear. It was decided that at the next visit the repeater would be opened, checked for any obvious problems and assuming none were found a new set of batteries would be installed in it.

9.1.4 Third visit - 28th May 2012

On the 28th of May 2012 a third visit was made. On this visit both network maintenance activities and propagation measurements through the crop near ground level were undertaken.

The repeater node was removed from the field and opened. A substantial amount of water was found inside and there was noticeable corrosion on the range extender board. It is suspected that the water leaked in through the lid or connector seals. The problems with the repeater node clearly could not be fixed on site. Therefore it was brought back to the University. After it was returned to the University the insides of the repeater were dried out, the range extender was replaced and silicone grease was added to the connector and lid seals before reassembly. The repeater was then tested and deemed to be working and it was decided to re-deploy it on the next visit.

To reduce the average power consumption of the the base station and hence the amount of time it was non operational due to discharge of the lead acid batteries it was decided to modify the base station to power down the mobile internet equipment outside working hours. To achieve this both hardware and software modifications were required and these modifications were deployed during this visit. The hardware modification was performed by replacing the power board in the base station with a modified version.

The propagation measurements through the crop near ground level were made using the surface stands and the same antennas used on the nodes. The tests were performed at distances of 0.5 m, 1 m, 1.5 m, 2 m, 2.5 m and 3 m with a frequency step of 10 MHz over a frequency range of 2.4 GHz to 2.5 GHz. Two separate sets of measurements were made through different sections of crop to allow for variability. The measurements were performed parallel to the edge of the crop approximately 1 m from the edge.

By this visit many of the flowers had fallen off the rapeseed plants leaving the crop looking mostly green rather than the yellow it had been while in full flower. A photograph of the crop during this visit is shown in Figure 9.6

Figure 9.6: Photograph of crop and high level antennas during third visit to farm.

9.1.5 Fourth visit - 14th June 2012

On the 14th of June 2012 a fourth visit was made to the farm. On this visit both network maintenance activities and propagation measurements around the crop canopy were undertaken.

During this visit the repeater node was reinstalled. Unfortunately on returning to the University and downloading data it was discovered that the nodes behind the repeater had not reappeared on the network.

By this time it had been noticed while testing in the laboratory that sometimes after long periods without joining a network nodes ended up in a "crashed" state where they stop searching for the network, the cause of this issue was not known at the time [1]. Nodes encountered in this state in the laboratory always recovered after a power cycle. It was suspected that nodes in the field behind the repeater node had ended up in a similar crashed state after they were left searching for a network for weeks.

It was therefore decided that if nodes did not reappear before the next visit they would be removed from the ground, opened and reset by briefly removing the batteries. The batteries would also have their voltage checked and be replaced if needed.

During this visit high level propagation measurements were made. These were made at heights of 1 m (just below crop canopy), 1.5 m (just above crop canopy) and 2 m (well above crop canopy) and distances of 10 m, 20 m and 30 m over the frequency range 2.4 GHz to 2.5 GHz with a frequency step of 5 MHz. To make these measurements the telescopic stands and the same antennas used on the sensor nodes were used. The measurements were made diagonally across the crop with one antenna at the edge of the crop and one antenna at the edge of the first tractor line.

By this stage nearly all the yellow flowers had dropped off from the rapeseed leaving the crop looking totally green. A photograph of this is shown in Figure 9.7.

Figure 9.7: Photograph of crop during fourth visit to farm.

[1] Much later the author noticed a serious flaw in the code for reading the real time clock which is now believed to be the cause of this failure.

9.1.6 Reappearance of nodes 1 and 3

Node 1 reappeared on the network at about 11:00 on the 22nd of July 2012. It is unclear what triggered this reappearance but message loss rates remained fairly low. Node 3 also returned at 20:00.

9.1.7 Fifth visit - 26th July 2012

On the 26th of July 2012 a fifth visit was made to the farm. On this visit both network maintenance activities and propagation measurements through the crop near the ground were undertaken.

During this visit nodes 2,4,5 and 6 were reset and their batteries were checked and found to still be in good condition with voltages around 6 V. It was also discovered during the reset procedure that some of the nodes were wet inside but there was not really anything that could be done about this and therefore the nodes were reassembled in this wet state. Node 2 reappeared on the network briefly a couple of times before reappearing reliably a couple of days later on the 28th. Curiously at about the same time node 3 also returned to reliable operation. Node 4 never reappeared on the network. Nodes 5 and 6 reappeared briefly but did not return to reliable operation.

The propagation measurements through the crop near ground level were made using the surface stands and the same antennas used on the sensor nodes. The tests were performed at distances of 0.5 m, 1 m, 1.5 m, 2 m, 2.5 m and 3 m with a frequency step of 5 MHz over a frequency range of 2.4 GHz to 2.5 GHz. Two separate sets of measurements were made through different sections of crop to allow for variability. The measurements were performed parallel to the edge of the crop approximately 1 m from the edge.

By this time the crop was nearly ready for harvest and had dried out and changed colour from green to brown. A photograph of this is shown in Figure 9.8.

Figure 9.8: Photograph of crop during fifth visit to farm.

9.1.8 Sixth visit - 2nd August 2012

The sixth visit was made on the 2nd August 2012. This was the last visit before the crop was harvested and the nodes removed by the farmer. During this visit high level propagation measurements were made at heights of 1m, 1.5m and 2m and distances of 10m and 20m with a frequency step of 5 MHz over a frequency range of 2.4 GHz to 2.5 GHz. It had also been planned to make measurements at a distance of 30m as before but the session was cut short due to rain. To make these measurements the telescopic stands and the same antennas used on the sensor nodes were used. No photographs are available of the crop during this visit but it did not appear to have changed significantly from the previous visit.

According to the farmer the crop was harvested soon after this visit, the farmer said he would leave the area around the nodes unharvested to avoid damaging the nodes. Later the farmer removed the equipment as he wanted to plant a new crop in the field.

9.1.9 Seventh visit - 27th June 2013

The seventh visit was made nearly a year later on 27th June 2013. A new crop was now growing in the field. The farmer had also removed the wireless sensor network equipment from the field.

The purpose of this visit was twofold. Firstly to retrieve equipment and data. Secondly to collect measurements in open space with the same antenna set-ups as used in the crop measurements as a baseline for comparison. This was done at the Farm as we could not arrange access to a suitable area for doing it at the University and being in the countryside reduces the possibility of interference from other equipment operating in the band.

During this visit measurements near ground level were collected, these were performed at distances of 0.5 m, 1 m, 1.5 m, 2 m, 2.5 m and 3 m with a frequency step of 5 MHz over a frequency range of 2.4 GHz to 2.5 GHz. As before the surface stands and the same antennas used on the sensor nodes were used.

It had also been intended to collect high level baseline measurements during this visit but rain stopped measurement activities before this could be done.

When the base station was opened substantial water was found inside. Fortunately the SD card survived and the final recordings from the network were recovered from it.

9.1.10 Eighth visit - 13 August 2013

The eighth and final visit to the farm was made on 13 August 2013.

During this visit high level measurements were made with no crop present for comparison purposes. As before these were made at heights of 1 m (just below crop canopy), 1.5 m (just above crop canopy) and 2 m (well above crop canopy) and distances of 10 m, 20 m and 30 m over the frequency range 2.4 GHz to 2.5 GHz with a frequency step of 5 MHz. To make these measurements the telescopic stands and the same antennas used on the sensor nodes were used.

9.2 Results and analysis of propagation measurements

9.2.1 Low level measurements in crop

The low level measurements were taken through the crop with the same antennas used for the deployment mounted on the surface stands. Measurements were taken with separation distances of 0.5 m, 1 m, 1.5 m, 2 m, 2.5 m and 3 m. Measurements were taken on two occasions, the 28th May 2012 and the 26th July 2012. On the 28th May 2012 Measurements were made with a frequency step of 10 MHz over a frequency range of 2.4 GHz to 2.5 GHz. On the 26th July 2012 the measurement step was reduced to 5 MHz increasing the number of data points.

The intent was that during each visit the measurements would be taken twice through different areas of crop. Unfortunately the first set of 1 m measurements during the session on the 28th May 2012 was accidentally overwritten.

Additionally a set of comparison measurements were taken on 27th June 2013 with the same measurement equipment, same antennas and stands and the same separation distances but with the antenna stands located on a farm track near the field with no obstructions between the antennas.

Each set of propagation measurements through the crop is presented in two forms. Firstly the measurements are presented as the "gain" between antenna connections, that is the power (in decibel units) at the receive antenna connector minus the power at the transmit antenna connector. Since a pair of antennas is a passive system the "gain" between antenna connections will always be negative indicating a loss. Secondly the measurements are presented as a comparison to measurements taken with no obstructions in an attempt to isolate the loss caused by passing through the crop from the normal losses of a system with basic antennas near ground level. This comparison was performed by subtracting the received power with no crop present from the received power with the crop present. In the comparative measurements a negative value indicates that the presence of the crop reduced the signal level (the expected result) and a positive value indicates that the presence of the crop increased the signal level (an unexpected result).

For the measurements through the crop there was felt to be little chance of the presence of people affecting measurements as the antennas were placed within the crop. However there was a real possibility that the results could be affected by crop damage caused by placing and removing the antennas and while efforts were made to place the antennas such that the direct path between them was through undisturbed crop this was sometimes difficult to achieve. For the unobstructed measurements, used for comparison, nobody walked directly between the antennas during the measurements but it was necessary to walk past the set-up to check on the status of the equipment. If this walking past had substantially impacted the results it is felt that the impact would have been in the form of isolated outlying points, no such points were observed.

As can be seen from the reflection measurements in Section 4.3 the crude monopoles used for the measurements and deployment are highly sensitive to the surrounding environment. As such the author does not reasonably believe it is possible to separate the effects of the antenna from those of the environment.

The measurements are presented in Figures 9.9 to 9.16.

Surprisingly the 0.5 m results from tests through the crop actually had less loss than the 0.5 m unobstructed tests, sometimes significantly less. The same effect was also seen with some but not all of the 1 m tests. The author suspects that over such short distances reflections from the crop and scattering of signals are aiding transmission more than the crop is attenuating the signal. The 0.5 m and 1 m unobstructed tests also showed a noticeable periodicity with frequency which is visible in the comparison graphs.

As can be seen from the reflection measurements in Section 4.3 these crude monopoles are highly sensitive to the surrounding environment. As such the author does not believe that measurements of gain or reflection of the antennas in free space would be applicable to operation within a crop environment. It would be possible to compare different antennas within the crop environment but this was not done.

In most cases loss increases with distance though occasionally a shorter distance measurement set showed a higher loss than a corresponding longer distance measurement set. The measurements show that the loss at 3 m on the 28th of May varied from approximately 90 dB to approximately 115 dB. As discussed in section 7.4 even with range extenders on both ends of the link, the limit for reliable communication with the transceivers used in the project is approximately 105 dB of loss. It is possible that alternative transceivers may tolerate slightly higher losses but the author still feels it is unlikely that distances much greater than this could be operated over reliably with this type and location of antennas.

Often the two tests for the same date and distance show very different results. The author attributes this to differences in foliage density both natural and caused by working in the crop. Rapeseed plants tangle together such that any attempt to move through the crop and place or remove equipment inevitably results in damage. Curiously the amount of variation with frequency also varied considerably between tests with some tests showing little variation with frequency and others varying by as much as 10 dB.

To further investigate the data and to compare results from the two dates it was decided to average the crop attenuation measurements over frequency and (where possible) over the two tests on a given date giving a single value for each combination of date and distance. As expected after this averaging the results clearly showed greater loss on the 28th of May than on the 26th of July.

It has been suggested in the literature [66] that models of the form shown in Equation 9.1 where L is loss from the foliage in dB, f is frequency (in GHz), d is distance in vegetation (in metres) and a b and c are constants of the model can be used to model path loss in vegetation with various constants being proposed for this form of model. Proposed values for a range from 0.2 to 15.6, proposed values for b range from -0.009 to 0.39 and proposed values for c range from 0.25 to 1.

$$L = af^b d^c \tag{9.1}$$

Since the frequency range is small the author feels it is unlikely to have a significant impact on the loss equation. It is also clear that there is an offset in the author's data with the

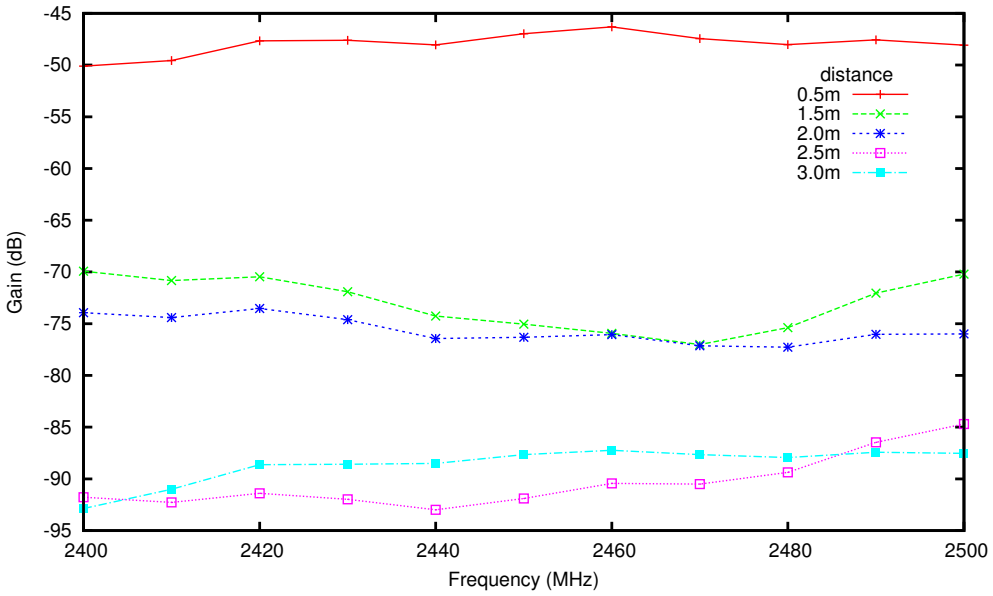


Figure 9.9: First set of measurements through crop on 28th May 2012, measured between antenna terminals.

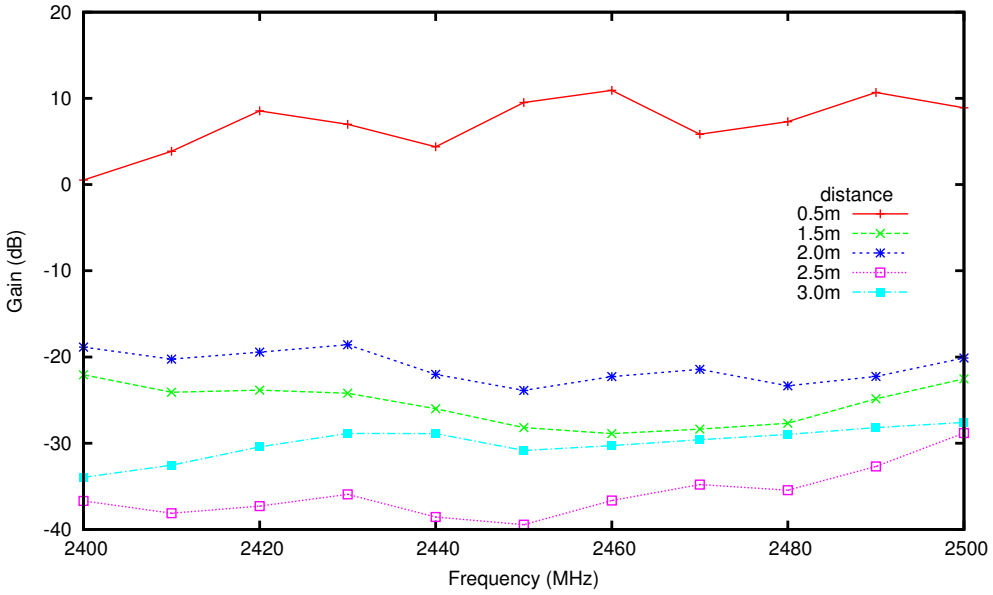


Figure 9.10: First set of measurements through crop on 28th May 2012, compared to unobstructed measurements.

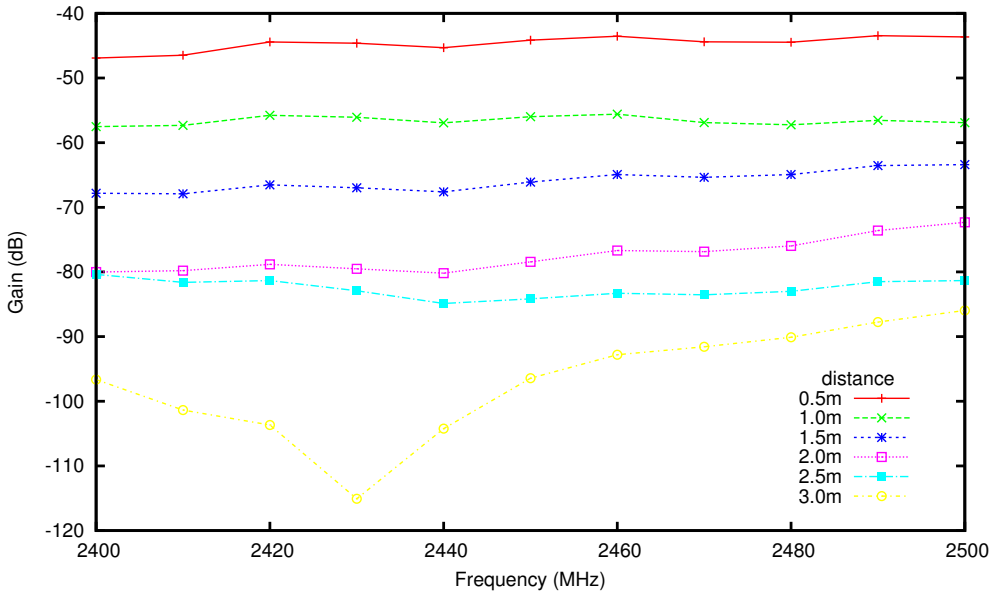


Figure 9.11: Second set of measurements through crop on 28th May 2012, measured between antenna terminals.

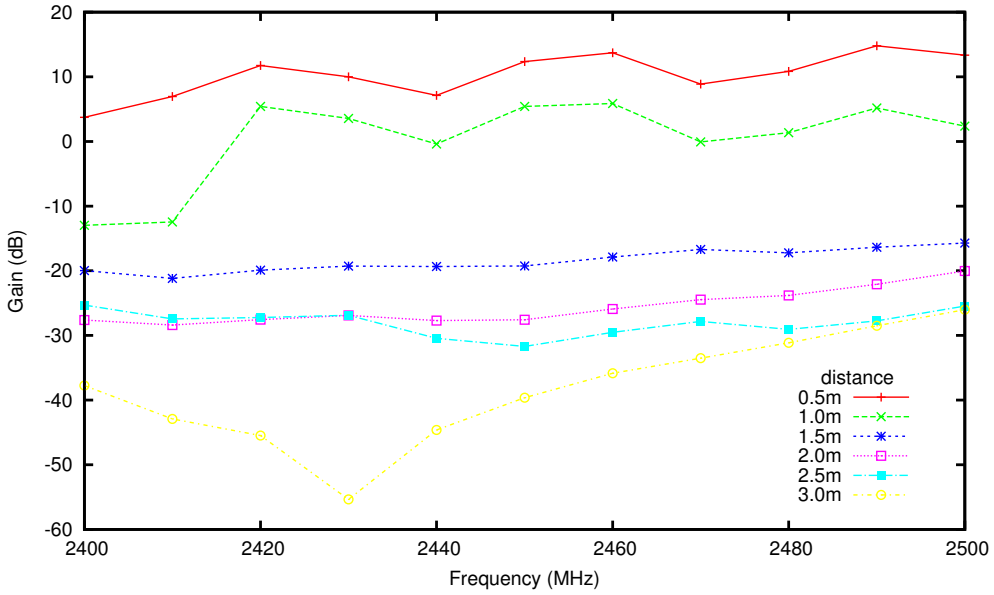


Figure 9.12: Second set of measurements through crop on 28th May 2012, compared to unobstructed measurements.

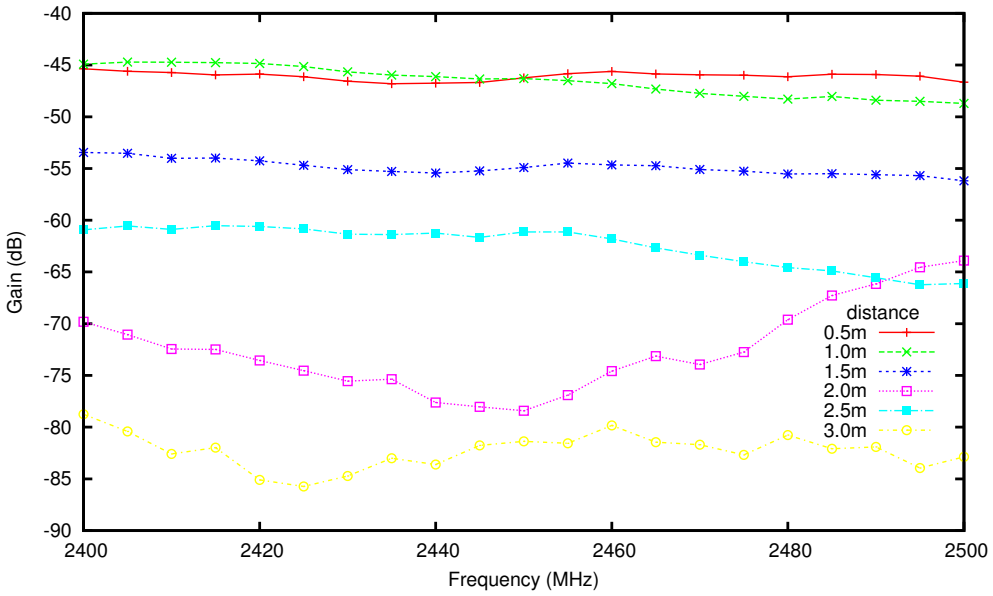


Figure 9.13: First set of measurements through crop on 26th July 2012, measured between antenna terminals.

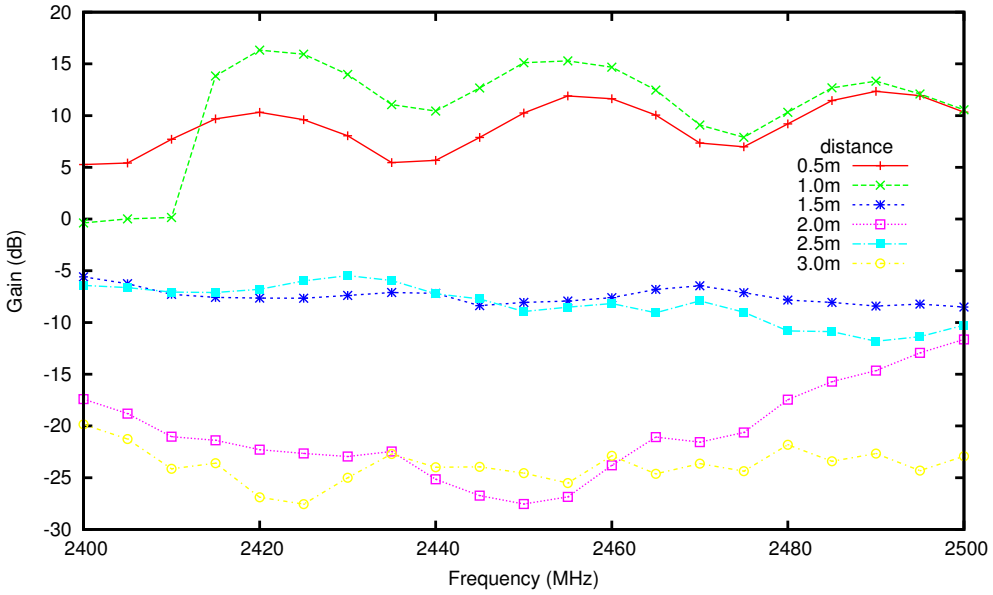


Figure 9.14: First set of measurements through crop on 26th July 2012, compared to unobstructed measurements.

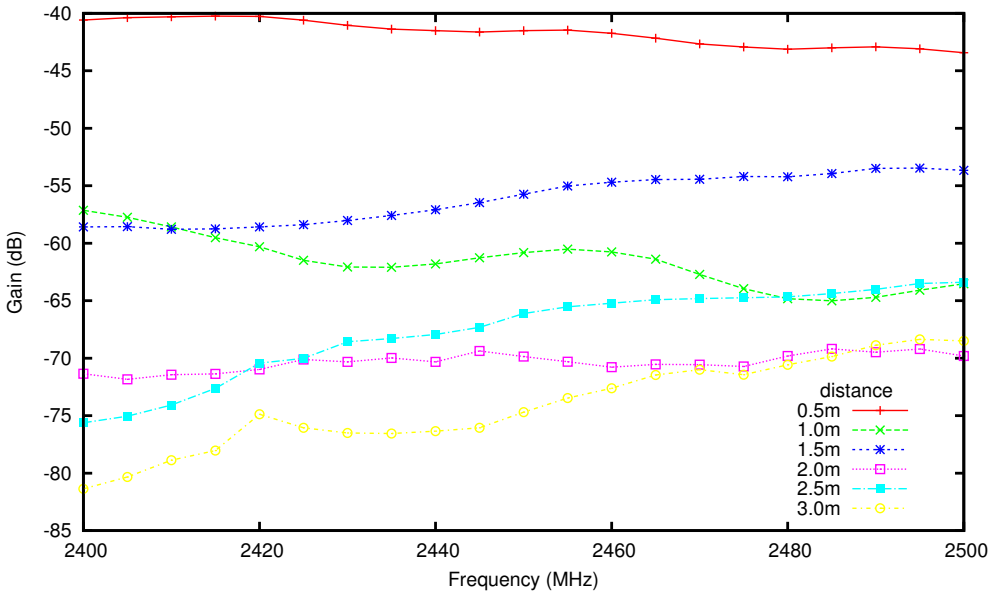


Figure 9.15: Second set of measurements through crop on 26th July 2012, measured between antenna terminals.

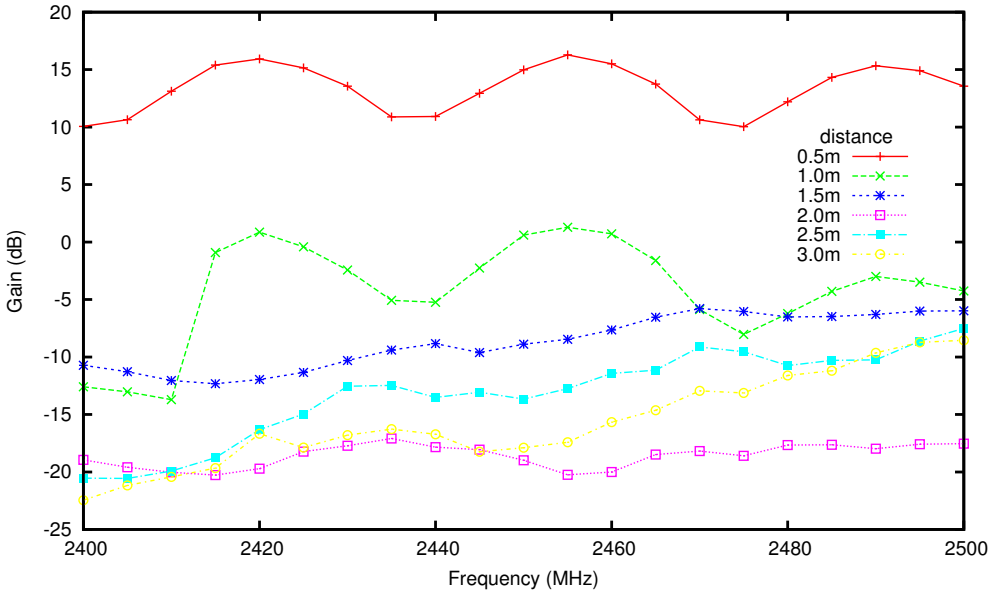


Figure 9.16: Second set of measurements through crop on 26th July 2012, compared to unobstructed measurements.

attenuation at short distances being lower than it was in the unobstructed test. Based on this a model of the form shown in Equation 9.2 where G is the "gain"[2] in dB from passing through the foliage, d is distance in the crop in metres and a, b and c are constants of the model was considered.

$$G = ad^b + c \tag{9.2}$$

The least squares fitting functionality in Matlab was used to fit this model to the data. First a version of the model where b was forced to be equal to 1 (making the equation linear) was fitted. This was then used as an initial condition to fit a version of the model where all three parameters were allowed to vary. Matlab returns the sum of the squares of the residual error as an indication of the quality of the fit. By dividing this by the number of data points and then taking the square root, the root mean square (RMS) of the residual error in the fit can be obtained.

Unfortunately when fitting the model in Equation 9.2 with all three constants chosen by the fitter produced constants that the author considers implausible with extremely high values for a and c and extremely low values for b. It appears that the fitter has fitted to the noise in this case. The author does not consider the difference in RMS error of less than 2 dB between the two equations to be significant compared to likely experimental variations. The results are shown in Table 9.1 with calculated values shown to 3 significant figures.

The measurements averaged over frequency, along with the results of the two different fits are plotted in Figure 9.17.

Date	Type	a	b	c	RMS error
28th May 2013	Linear	-15.1	1	12.0	4.21
28th May 2013	Power	-108	0.193	103	3.06
26th June 2013	Linear	-10.84	1	12.3	4.62
26th June 2013	Power	934	0.0170	934	3.06

Table 9.1: Parameters obtained from fitting equations to propagation data using Matlab

9.2.2 High level measurements close to crop canopy and above crop

The high level measurements were made with the same antennas used on the nodes during deployment. These were attached to the telescopic stands described in Section 6.3. Measurements were made with the radiating elements of the antennas 1 m, 1.5 m and 2 m off the ground and with distances between the antennas of 10 m, 20 m and 30 m.

Measurements were made on two occasions. The first was the 14th of June 2012 and the second was on the 2nd of August 2012. During the visit on the 14th of June 2012 the full set of measurements described above was made. During the visit on the 2nd of August 2012 no measurements were made at 30 m separation due to rain.

The height of the crop was typically 1.3 m on the 14th of June 2012 and did not noticeably change between the two sessions. This meant that an antenna 1 m from the ground was just

[2] while expressed as a gain this will usually be a negative number indicating loss

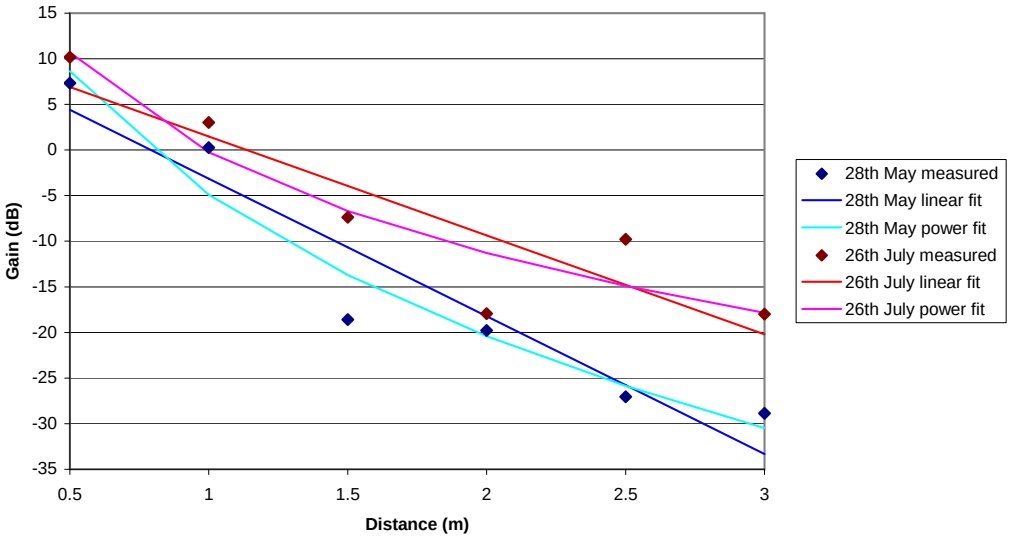


Figure 9.17: Propagation measurements in crop compared to unobstructed measurements and averaged over frequency and where possible multiple measurement runs. With Matlab Fits.

below the crop canopy. An antenna 1.5 m above the ground was just above the bulk of the crop canopy and an antenna 2 m above the ground was well above the crop canopy.

Results are plotted in Figures 9.18 to 9.29 and a summary of the highest and lowest measurements for each combination of date, height and distance are tabulated in Tables 9.2 to 9.5.

As before each set of propagation measurements through the crop is presented in two forms. Firstly the measurements are presented as the "gain" between antenna connections, that is the power (in decibel units) at the receive antenna connector minus the power at the transmit antenna connector. Secondly the measurements are presented as a comparision to measurements taken with no crop present in an attempt to isolate the loss caused by passing through the crop from the normal losses of a system with basic antennas at the same height above ground. As before a negative value in the comparision measurements indicates that the presence of the crop reduced the signal level and a positive value indicates that the presence of the crop increased the signal level.

For the measurements with a path over the crop there was nobody in the crop during measuremenents and hence little chance of the presence of people affecting measurements. For the unobstructed measurements, used for comparison, nobody walked directly between the antennas during the measurements but it was necessary to walk past the set-up to check on the status of the equipment. If this walking past had substantially impacted the results it is felt that the impact would have been in the form of isolated outlying points, no such points were observed.

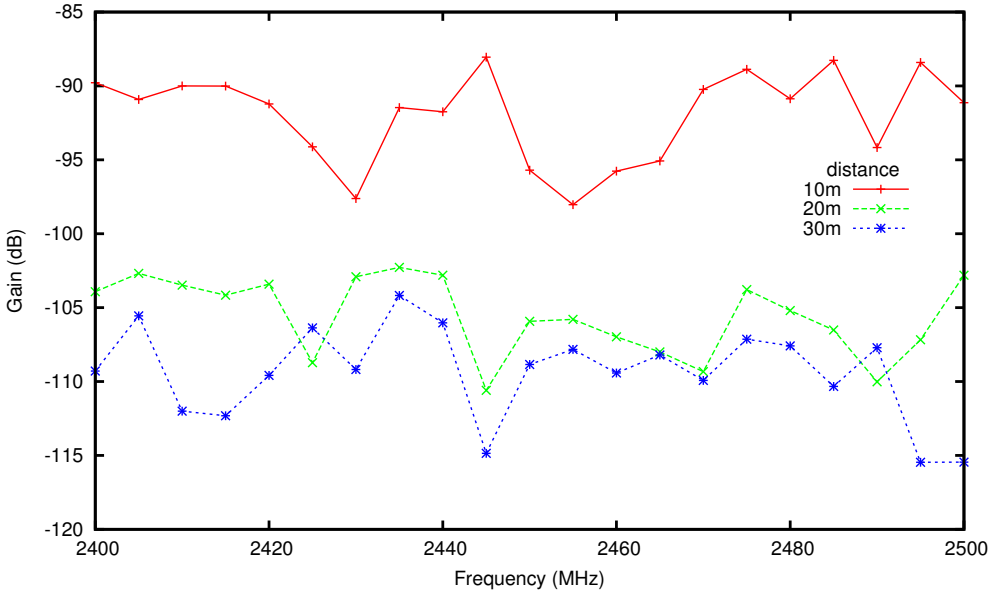


Figure 9.18: Measurements at 1 m above ground in crop on 14th June 2012, measured between antenna terminals.

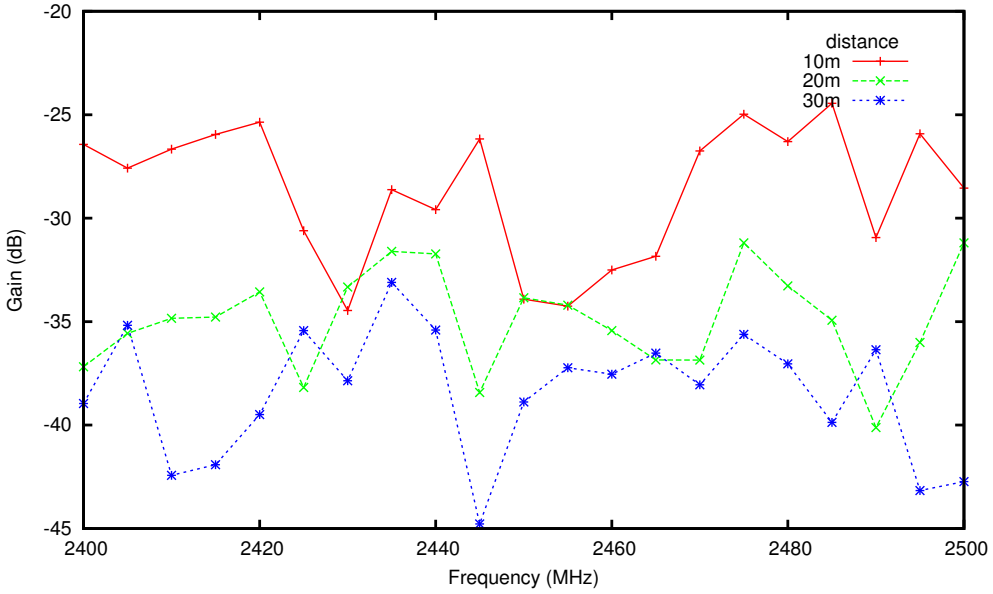


Figure 9.19: Measurements at 1 m above ground in crop on 14th June 2012, compared to measurements with no crop present.

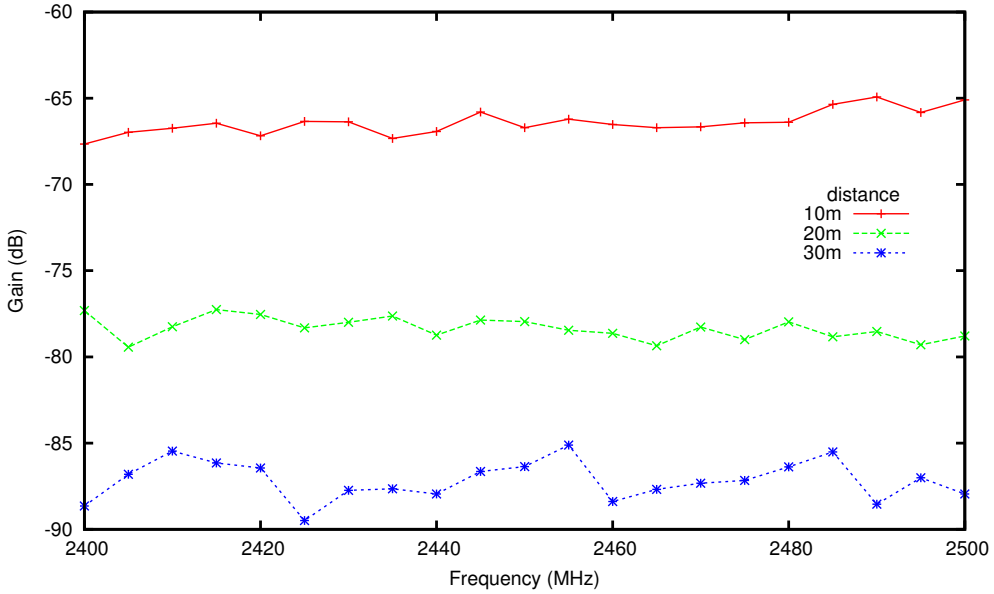


Figure 9.20: Measurements at 1.5 m above ground over crop on 14th June 2012, measured between antenna terminals.

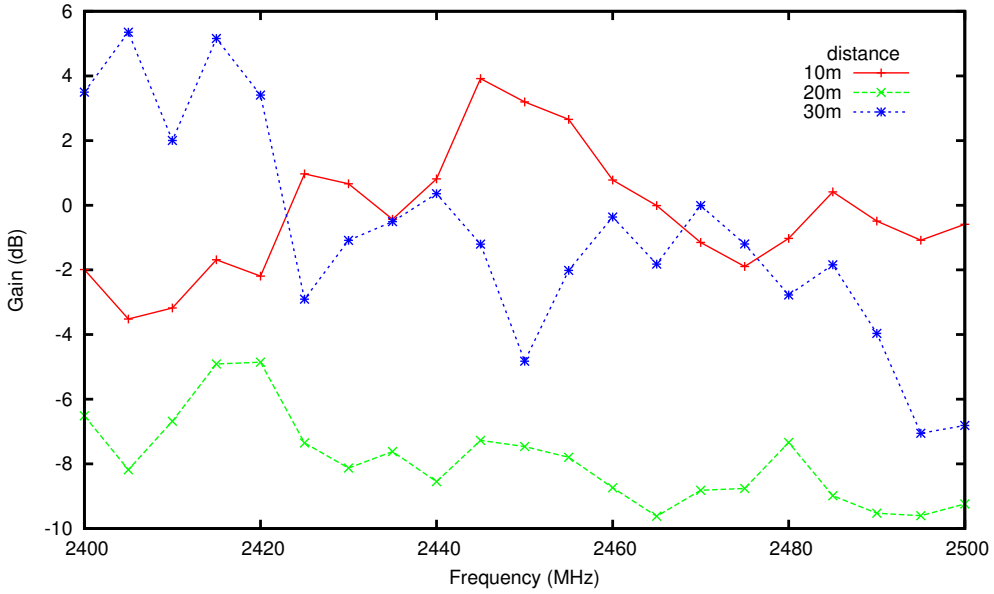


Figure 9.21: Measurements at 1.5 m above ground over crop on 14th June 2012, compared to measurements with no crop present.

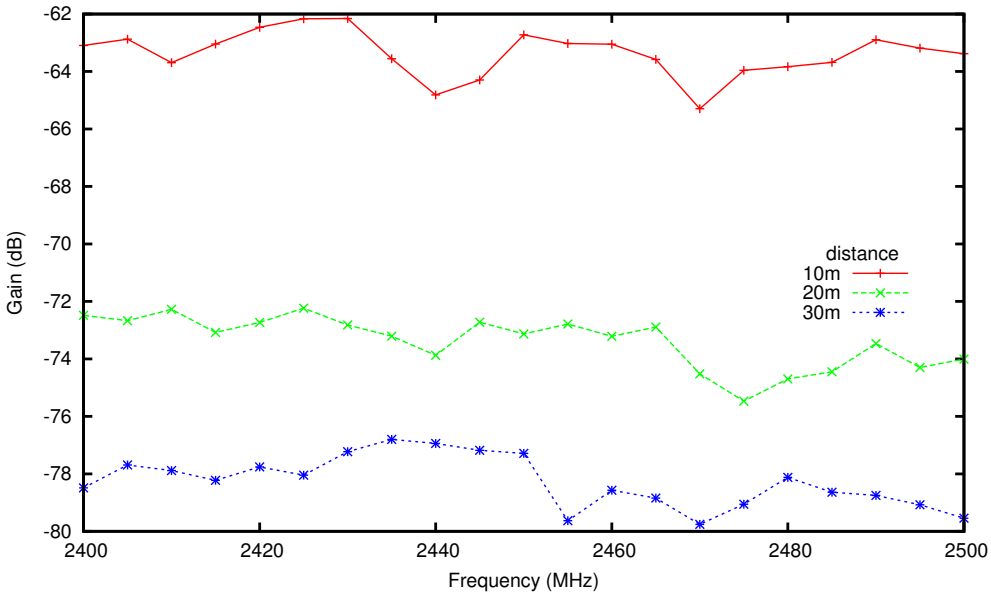


Figure 9.22: Measurements at 2 m above ground over crop on 14th June 2012, measured between antenna terminals.

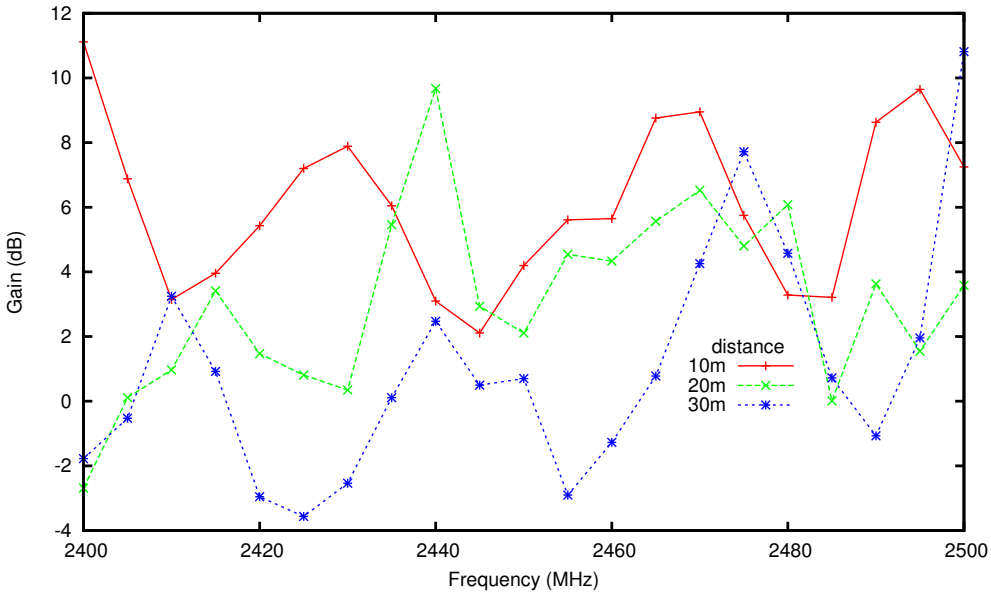


Figure 9.23: Measurements at 2 m above ground over crop on 14th June 2012, compared to measurements with no crop present.

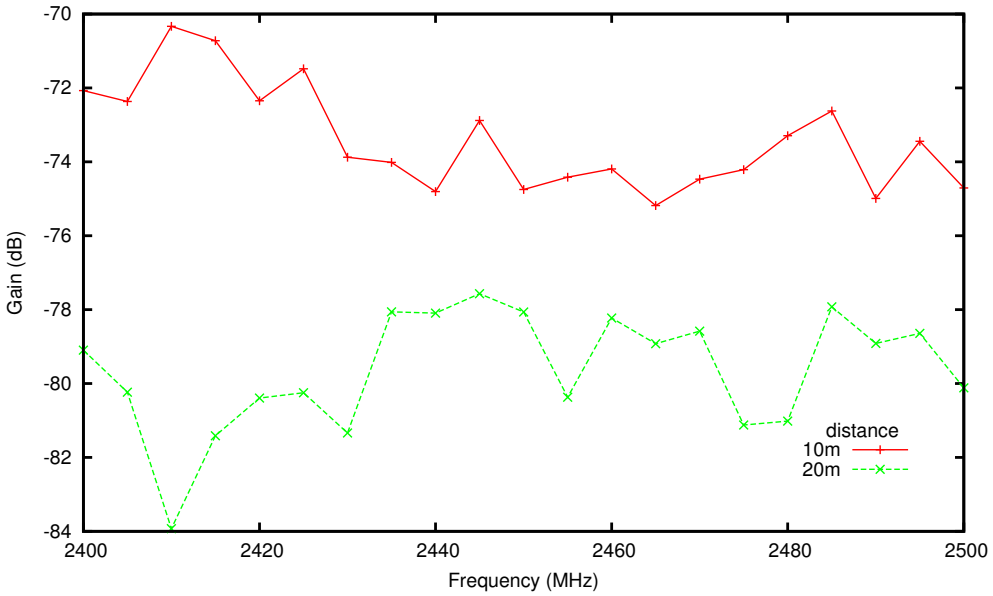


Figure 9.24: Measurements at 1 m above ground in crop on 2nd August 2012, measured between antenna terminals.

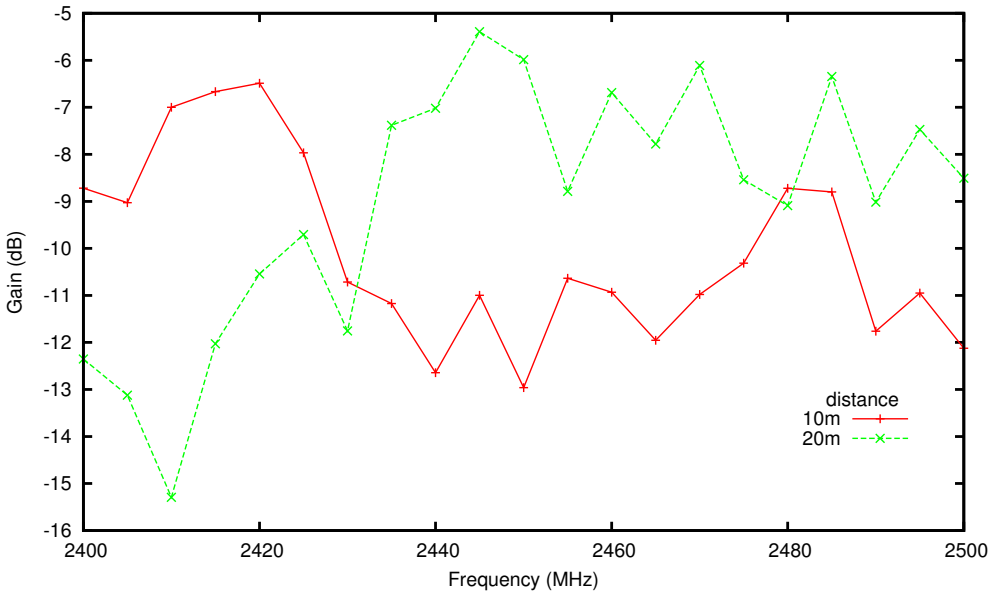


Figure 9.25: Measurements at 1 m above ground in crop on 2nd August 2012, compared to measurements with no crop present.

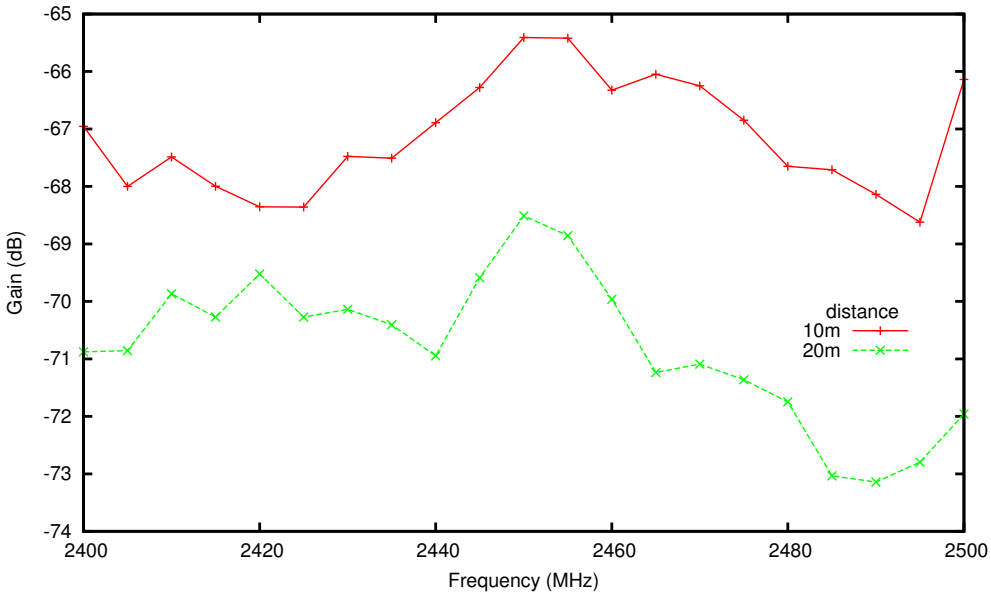


Figure 9.26: Measurements at 1.5 m above ground over crop on 2nd August 2012, measured between antenna terminals.

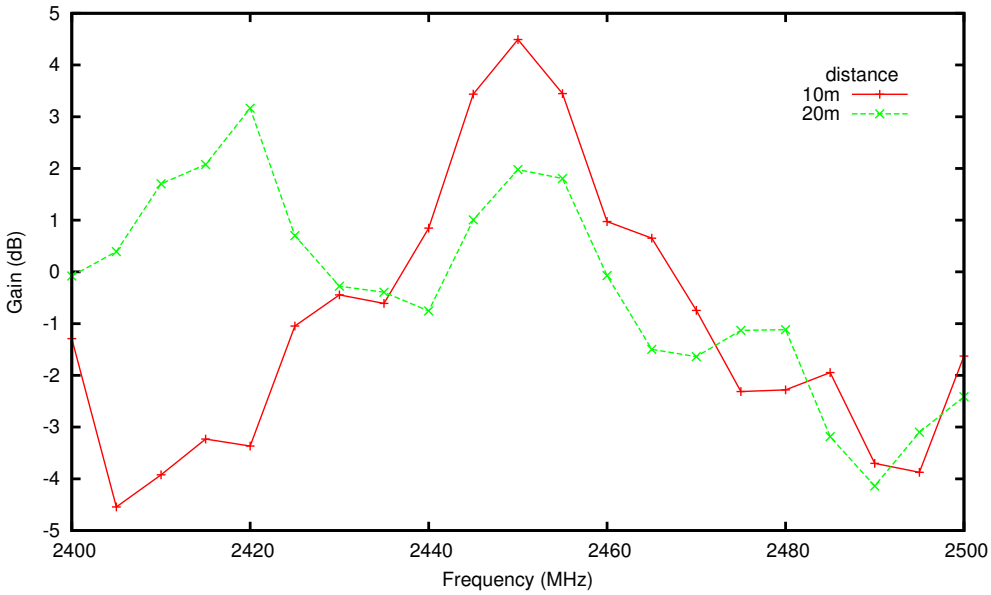


Figure 9.27: Measurements at 1.5 m above ground over crop on 2nd August 2012, compared to measurements with no crop present.

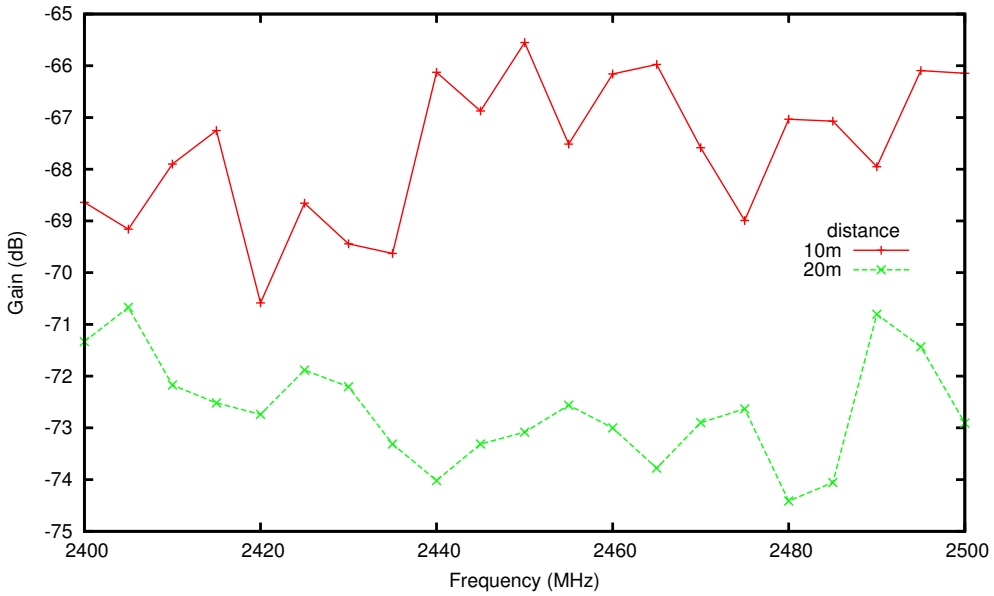


Figure 9.28: Measurements at 2 m above ground over crop on 2nd August 2012, measured between antenna terminals.

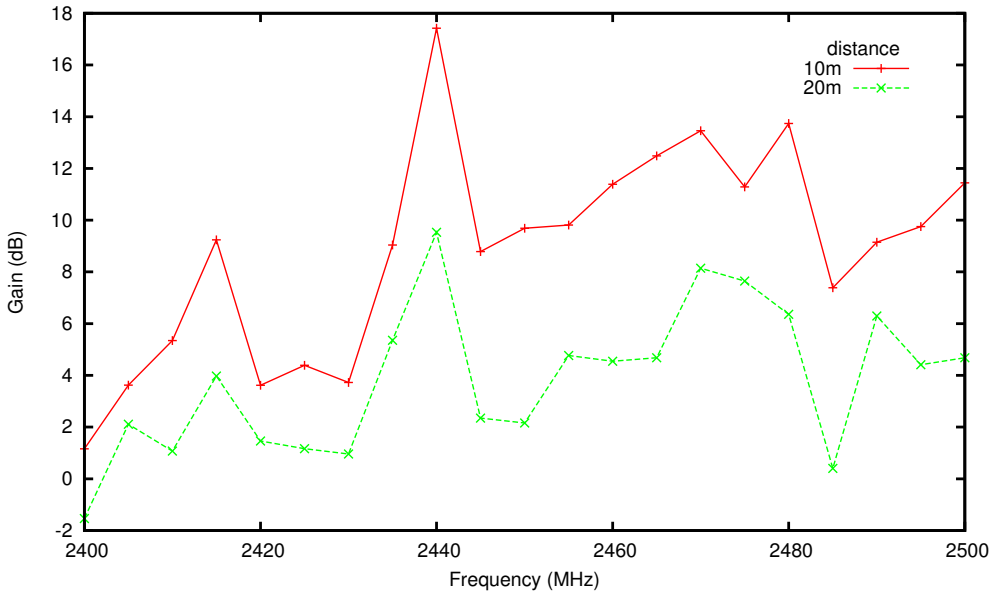


Figure 9.29: Measurements at 2 m above ground over crop on 2nd August 2012, compared to measurements with no crop present.

Distance	10 m		20 m		30 m	
Height	Min	Max	Min	Max	Min	Max
1 m	-98 dB	-88 dB	-111 dB	-102 dB	-115 dB	-104 dB
1.5 m	-68 dB	-65 dB	-79 dB	-77 dB	-89 dB	-85 dB
2 m	-65 dB	-62 dB	-75 dB	-72 dB	-79 dB	-76 dB

Table 9.2: Summary of measurements on 14th of June 2012, measured between antenna terminals

Distance	10 m		20 m		30 m	
Height	Min	Max	Min	Max	Min	Max
1 m	-34 dB	-24 dB	-40 dB	-31 dB	-44 dB	-33 dB
1.5 m	-5 dB	4 dB	-10 dB	-5 dB	-7 dB	5 dB
2 m	2 dB	11 dB	-3 dB	10 dB	-4 dB	11 dB

Table 9.3: Summary of measurements on 14th of June 2012, compared to unobstructed measurements

Distance	10 m		20 m	
Height	Min	Max	Min	Max
1 m	-75 dB	-70 dB	-84 dB	-78 dB
1.5 m	-69 dB	-65 dB	-73 dB	-69 dB
2 m	-71 dB	-66 dB	-74 dB	-71 dB

Table 9.4: Summary of measurements on 2nd of August 2012, measured between antenna terminals

Distance	10 m		20 m	
Height	Min	Max	Min	Max
1 m	-13 dB	-6 dB	-15 dB	-5 dB
1.5 m	-5 dB	4 dB	-4 dB	3 dB
2 m	1 dB	17 dB	-2 dB	10 dB

Table 9.5: Summary of measurements on 2nd of August 2012, compared to unobstructed measurements

Based on these results and the link budget measurements discussed in Section 7.4, transmission under the conditions seen in June should be feasible at 1 m antenna height and 10 m separation with range extenders but working at a height of 1 m and a distance of 20 m is likely to pose reliability problems even with the use of range extenders at both ends. Curiously while the losses at 20 m are substantially worse than at 10 m there is far less difference between the 20 m and 30 m results.

Increasing the antenna height to 1.5 m reduced the losses substantially. The highest observed signal loss for each distance improved by approximately 30 dB and the losses at 1.5 m height and 30 m distance were lower than losses at 1 m height and 10 m distance. Increasing the height of the antennas above the ground to 2 m produced further improvements of approximately 7 dB.

The comparison between measurements with and without crop reveals that substantial loss is being caused by the crop when the antennas are 1 m above the ground. With the antennas 1.5 m and 2 m above the ground the loss with the crop present is comparable to and in some cases lower than the loss with no crop present.

The results from August show considerably lower losses at 1 m height than in June. Again the results at 1.5 m and 2 m height are comparable to those with no crop present.

9.3 Results and analysis of deployment

The surface water presence sensors were fouled with soil during installation. This was due to misunderstandings between the people involved. Further due to their closeness to the ground it was very likely that even if they had not been fouled initially they would have become fouled during the deployment. The size and shape of the holes in which the sensor nodes were installed made it impractical to control the packing of the soil into which the soil resistance probes were inserted. The data from the soil resistance sensors showed large variations between the nodes, with some nodes stuck in saturation. As a result of these difficulties with the soil resistance and water presence sensors it was felt that further analysis of the data from them would not produce any meaningful results.

As a result there were three interesting things to investigate. The measurements from the temperature sensors, the proportion of messages from each node that were lost at different times and the measurements of received signal strength and "link quality" from the RF transceiver in the base station. While analysing the number of messages lost from each node at different times, the author also noticed some issues related to the clocks in the system becoming unsynchronised.

The temperature measurements and RSSI readings were reported as individual data points and as such it is not possible to calculate statistical error bars for them. While the message loss rates were calculated through an averaging process the author does not belive that calculating statistical error bars would be meaningful due to the low level of independence in the measurements.

The important events in the operation of the network described in detail in Section 9.1 are summarised here in table 9.6. The final layout of sensor nodes after the substitution of non-functional nodes performed on the second visit to the farm is shown in Figure 9.30.

Visit	Date	Actions/events	Crop status
1	15-Mar	Initial deployment	Scattered small plants
	11-Apr	Repeater node failed	
2	3-May	Nodes 10 and 12 replaced by nodes 13 and 14 High level antennas for repeater raised	Full flower
3	28-May	Repeater node removed and returned to the university	Still flowering but flowers noticably less dense
4	14-Jun	Repeater node re-deployed but nodes behind it do not immediately reappear	Lush and green with most flowers gone
	22-Jul	Nodes 1 and 3 reappear	
5	26-Jul	Nodes 2, 4, 5 and 6 reset	Dry brown
6	2-Aug	Last visit before equipment was removed by farmer	Dry brown

Table 9.6: Summary of important events in the operation of the network in 2012.

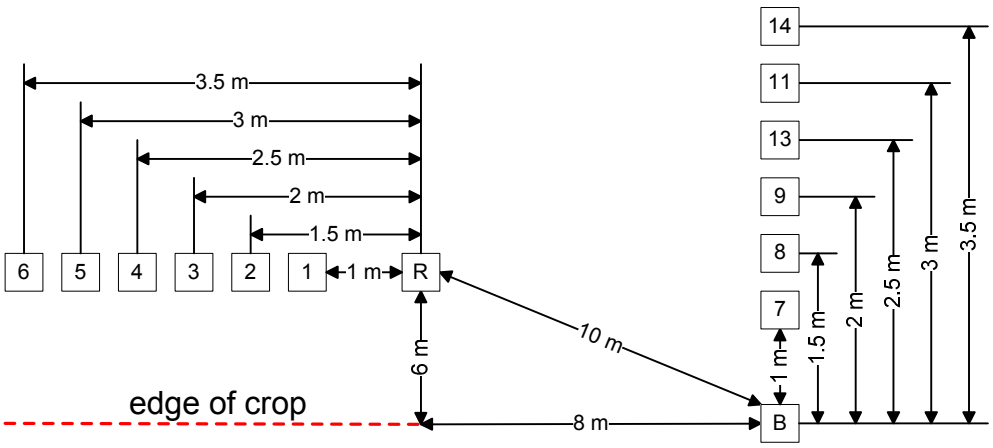


Figure 9.30: Diagram of layout of equipment in Tatton Dale Farm deployment. After node substitutions, B is the base station, R is the repeater and numbered boxes are end nodes.

9.3.1 Temperature readings

As mentioned previously each node has two temperature sensors. The first sensor measures the internal temperature of the node. Since the node is buried, it is believed this will reflect soil temperature. The second sensor is attached to the node's external antenna connector, as this is an exposed metal object it is believed this will reflect the air temperature within the crop (which may of course differ from the air temperature in the area in general). These sensors will be referred to as the "internal" and "external" sensors respectively.

The main question of interest is the difference in readings between the temperature sensors in nodes located closely together. The magnitude of these differences would indicate whether it would ever be desirable to locate nodes whose function was sensing temperature so close together in a crop research deployment[3].

It was decided to focus investigation on the four nodes that had been present and operational for the entire course of the deployment, that is nodes 7,8,9 and 11. All of these nodes were reporting directly to the base station and were located at distances from it ranging from 1 m to 3 m. Further it was decided to only take the data points where all four of these nodes had successfully reported.

To convert the ADC readings from ADC units to temperature values in degrees centigrade a scale factor and offset were applied. This scale factor and offset were based on values from the data sheets for the temperature sensor and microcontroller and the assumption that the system supply voltage was 3.3 V.

The data sheet for the temperature sensor [70] quotes a typical accuracy of $\pm 1°C$ but implies that to achieve this level of accuracy requires calibration of individual sensors. No calibration[4] measurements were available from individual sensors.

This data is plotted in Figures 9.31 and 9.32 with each point on the graph representing an individual reading from the sensor. However while these graphs show that the temperature results from the nodes are similar they do not give us useful numeric comparisons.

Therefore it was decided that the best approach was to normalise the temperature readings so that each sensor had the same mean and standard deviation. After normalising, the readings from different sensors could be compared more readily. The normalisation process went as follows. Normalisation was performed separately for the internal and external sensors.

1. Take the subset of time points and nodes mentioned above
2. For each time point in the above subset calculate the mean temperature across all nodes considered.
3. Calculate the mean and standard deviation for each node and for the "mean across nodes" calculated above.

[3] it had already become clear to the author that locating nodes so close together would be impractical in a commercial farm deployment

[4] it had been hoped to make calibration measurements on the sensors in the nodes after the nodes were retrieved from the field but time delays in arranging a visit to retrieve them meant this did not happen

4. Calculate a scale factor and offset to normalise the mean and standard deviation from each node to match the mean and standard deviation calculated from the mean across nodes.

The scale factors and offsets used are shown in tables 9.7 and 9.8 (note: the figures in the table are given to two decimal places, the figures used in the actual calculations were not deliberately rounded).

The data processing was done with a version of the processing software described in Section 7.10 that rounded message times to the nearest 10 minutes.

After normalisation the lowest value for each time point was subtracted from the highest value for each time point. This provides information of how much difference there is between the information from different sensors. The results of this are shown in Figures 9.33 and 9.34. The difference peaks at approximately 6 °C for the internal temperature sensor and approximately 10 °C for the external sensor. These differences are sufficient to potentially be significant to plants.

Node number	Scale factor	Offset (°C)
7	1.05	-0.11
8	1.02	0.01
9	0.94	0.51
11	0.88	0.91

Table 9.7: Scale factors and offsets used to normalise data from the internal temperature sensors

Node number	Scale factor	Offset (°C)
7	1.02	-0.38
8	0.98	0.62
9	0.95	0.35
11	0.98	0.26

Table 9.8: Scale factors and offsets used to normalise data from the external temperature sensors

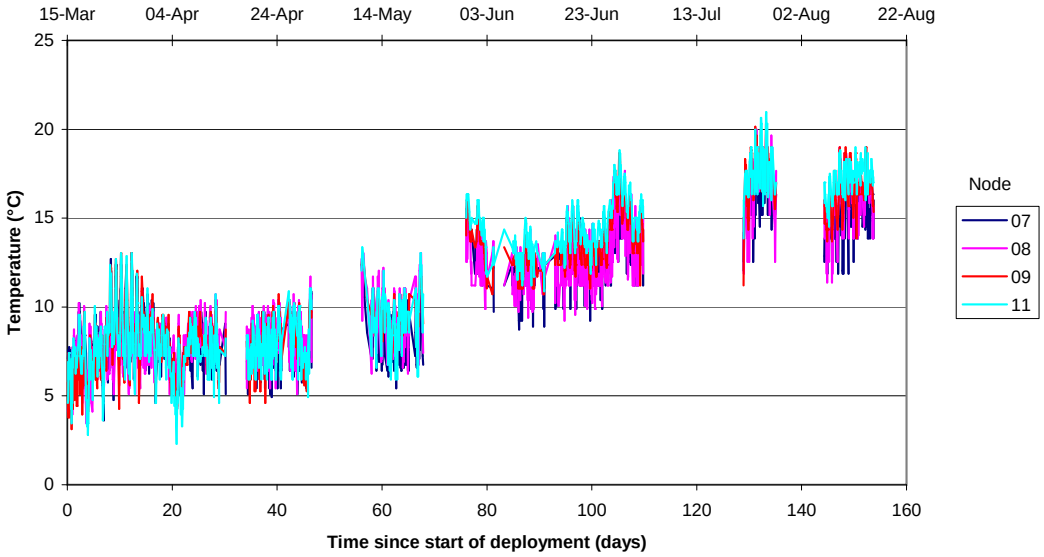


Figure 9.31: Measurements from internal temperature sensor before normalisation.

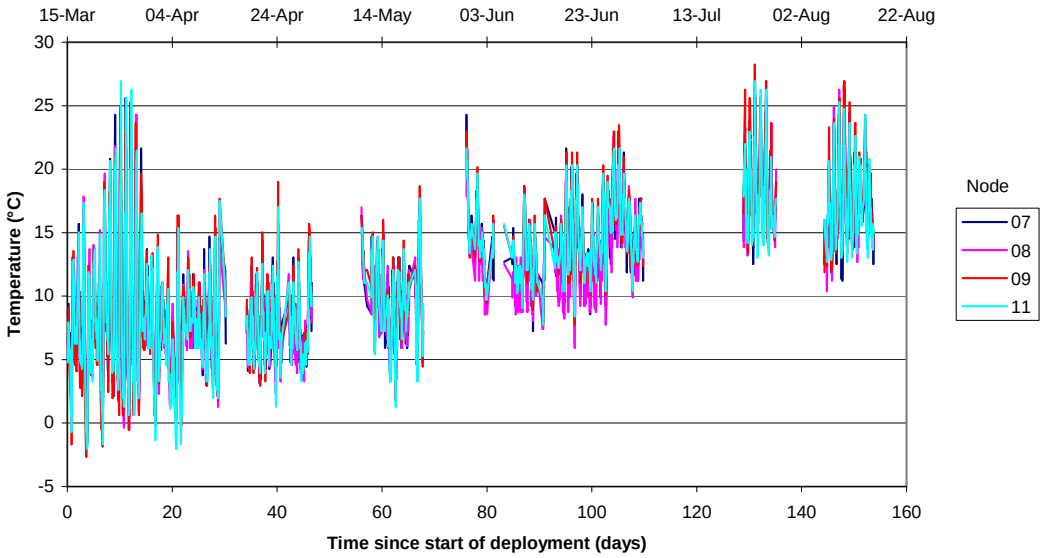


Figure 9.32: Measurements from external temperature sensor before normalisation

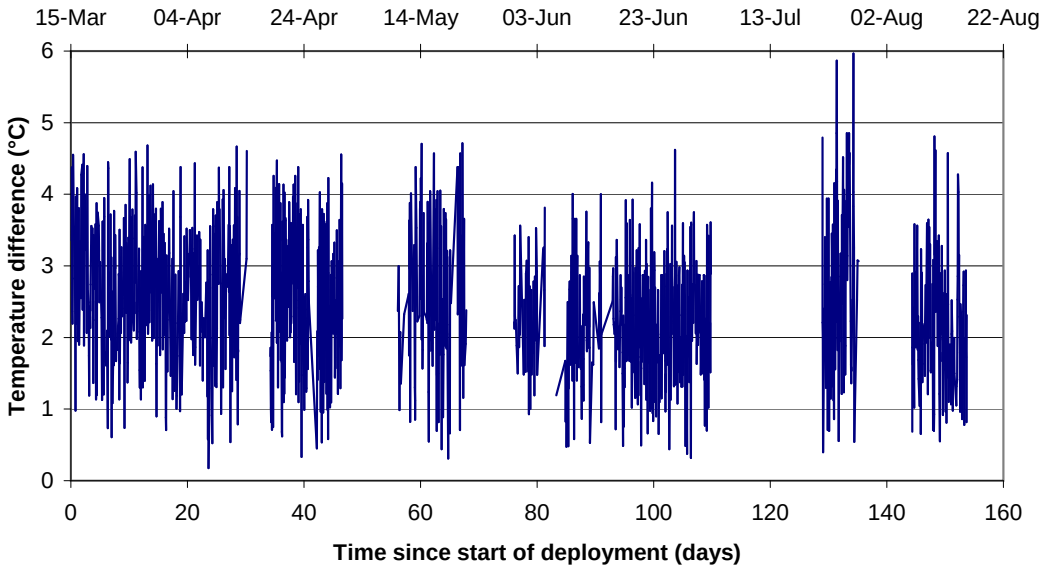


Figure 9.33: Difference between highest and lowest normalised temperature values for internal temperature sensor.

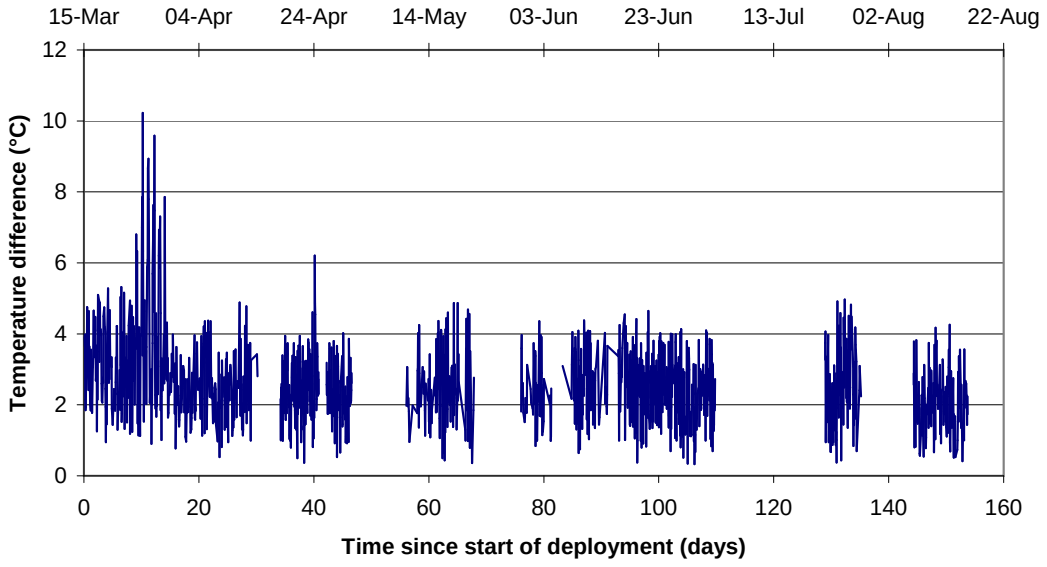


Figure 9.34: Difference between highest and lowest normalised temperature values for external temperature sensor

9.3.2 Message loss rates

To analyse the message loss rates of nodes, all time points that corresponded to an expected message (time points not corresponding to expected time points were ignored) and for which at least one node had successfully transmitted a message to the base station were considered. Time points where no nodes successfully transmitted a message were assumed to represent problems with the base station and were ignored (not counted as either success or failure). The data was processed with the revised version of the processing software described in Section 7.10 that rounded times to the nearest 30 minutes.

Message loss rates were calculated as 5-day averages. A period of 5 days was chosen to provide a reasonable number of points to average given that the nodes transmitted one message per hour. Since time points where no messages were received were ignored the number of points included in an average varied from 5 to 120. If no messages at expected time points were gathered during a given 5-day period then no output values were produced for that period. To provide better temporal resolution the time periods were overlapped with the centre of the averaging period moved in half day steps.

Due to network downtime there were a three gaps during which no messages were received from any node and hence no averages were calculated. Furthermore on either side of these gaps there are a number of time periods for which the average was calculated from less than 50 points and so was considered by the author to be unreliable. The first gap ran from 49.5 days from start of deployment to 53.5 days from start of deployment with unreliable data running from 47.5 days from start of deployment to 55 days from start of deployment. The second gap ran from 112.5 days from start of deployment to 126 days from start of deployment with unreliable data running from 110.5 days from start of deployment to 128.5 days from start of deployment. The third gap ran from 138 days from start of deployment to 141.5 days since start of deployment with unreliable data running from 136 days from start of deployment to 143.5 days since start of deployment.

In addition there were two other periods of network downtime that while not completely stopping the generation of averages did cause the number of data points in the average to drop below 50. The first such period of unreliable data ran from 31 days from start of deployment to 33.5 days from the start of deployment. The second ran from 70.5 days from the start of deployment to 72.5 days from the start of deployment.

As mentioned previously the nodes placed with the repeater largely stopped appearing in the data gathered by the base station after the 4am reading on 11 April 2012 (26.75 days from the start of the deployment) and this is attributed to technical problems with the repeater, not to any problems with the individual nodes. This means that the last message loss value that can be considered usable for the nodes placed with the repeater was the one centred on 24 days from the start of deployment. Some data was also gathered from these nodes at the very end of the deployment after the repeater was re-deployed but the author does not believe this data to be trustworthy due to the large number of problems encountered with these nodes. The results from these nodes are shown in Figure 9.35

During the period when the repeater was operational at the start of the deployment, the message loss rate for the four nodes closest to the repeater (at distances ranging from 1 m to

2.5 m) varied between 5 % and 31 %. This was higher than hoped but indicates the majority of messages were received successfully. Node 5 (3 m from the repeater) started with low loss but after the 9 day point packet loss rates started rising sharply reaching 92.5 % by the last 5 day period containing usable data. Node 6 (3.5 m) showed a very strange pattern starting off around 30 % loss, rising to 80 % then dropping back to less than 20 % before rising again towards the end.

For the nodes transmitting directly to the base station useful data was obtained for the entire duration of the deployment and is plotted in Figure 9.36. Additionally Figure 9.37 shows the same data but with averages generated from less than 50 time points removed.

Discounting time periods where the average was produced from less than 50 data points nodes 7 and 8 (1 m and 1.5 m showed less than 20 % message loss rates throughout the deployment. Node 9 showed slightly higher losses peaking at 26 %. When it was inroduced to the network node 13 (the replacement for node 10, 2.5 m from the base station) showed low loss (lower in-fact than nodes 8 and 9 which were closer to the base station) but then rose to nearly 37 % about 85 days from the start of deployment before falling again towards the end of the deployment. Curiously this maximum came later than the peak for more distant nodes.

Node 11 (3 m from the base station) showed low message loss initially rising to an almost total loss in the middle of the deployment with a maximum loss of 97.7 % in the period centred round 82.5 days from the start of the deployment. It then returned to low loss levels towards the end of the deployment as the crop dried out. After it was introduced to the network node 14 (the replacement for node 12, 3.5 m from the base statation) initially showed a loss of 50 %. It quickly shot up to 100 % and remained high for some time. Like node 11 the loss rate for node 14 also dropped at the end of the deployment. Both nodes 11 and 14 showed a very spiky message loss profile with periods of almost total loss and periods of much lower loss.

Interestingly the nodes reporting direct to the base station showed a significant reduction in message loss rates after the repeater failed. Indeed for nodes 7, 8 and 9 (the three nodes closest to the base station the highest loss seen was during the initial period. This suggests that collisions may have been a problem when all nodes were operational.

Based on the large jump in peak message loss rates from 37 % for node 13 (2.5 m from the base station) to 97.7 % for node 11 (3 m from the base station) the author came to the conclusion that if a deployment is expected to operate throughout the season then 2.5 m is approximately the limit for transmitting through the crop near ground level with the hardware and antenna arrangements used. As discussed previously the network used MRF24J40 [68] radio transceivers with a CC2591 range extenders [112] in the base station and repeater nodes but not in the end nodes. The range could be extended slightly by installing range extenders in the end nodes or finding a RF transceiver that can operate with a lower received signal level. However the author believes that while these measures may increase the operating distance by approximately 1 m, that operating near ground level in a rapeseed crop in the 2.4 GHz band is only practical if a very high node density is desired.

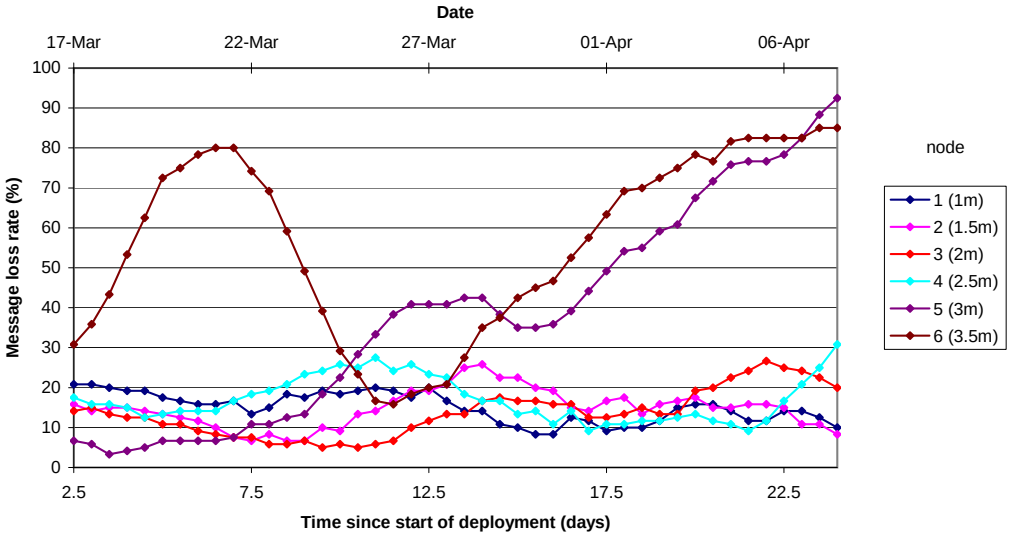


Figure 9.35: Message loss rates for nodes behind the repeater.

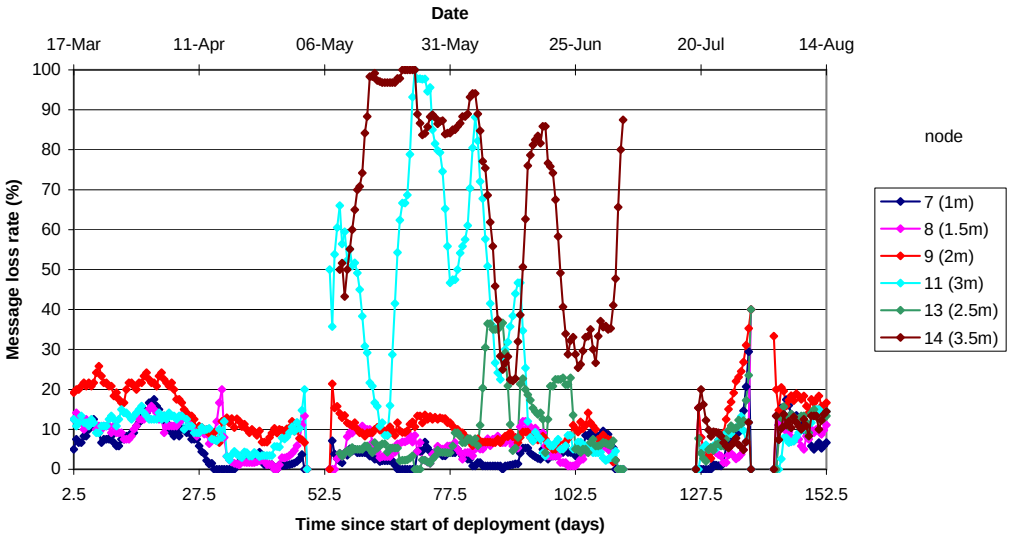


Figure 9.36: Message loss rates for nodes transmitting directly to the base station.

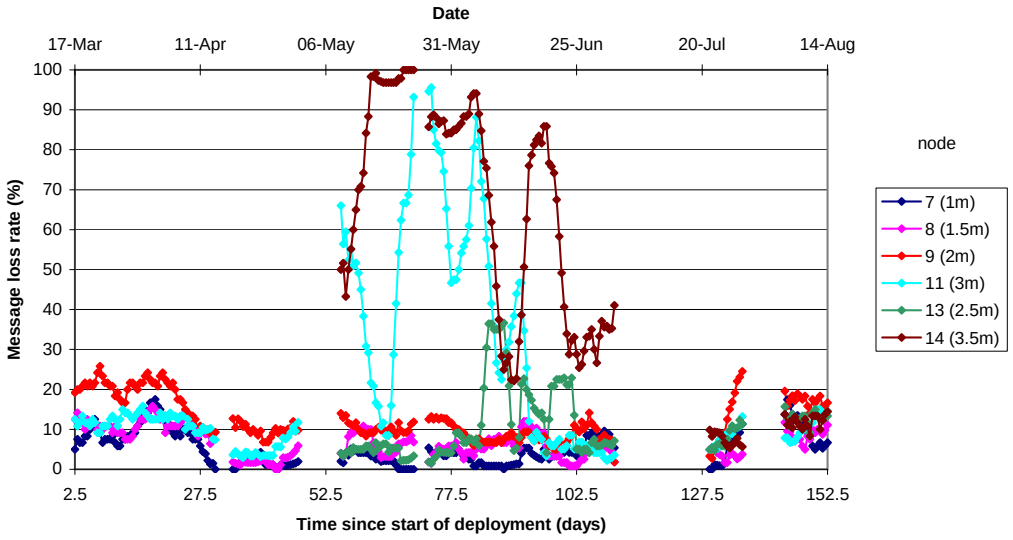


Figure 9.37: Message loss rates for nodes transmitting directly to the base station, averages generated from less than 50 points excluded.

9.3.3 RSSI and LQI readings

The MRF24J40 RF transceivers used in the nodes can provide a Received Signal Strength Indication (RSSI) and Link Quality Indication (LQI) for each packet received. These are saved for messages arriving at the base station and reflect the signal power and quality of the packet received at the base station. Therefore for packets delivered via a repeater they represent the strength of the link from repeater to base station, not the link from end node to repeater.

The received signal strength indication is provided by the MRF24J40 as a number between 0 and 255. The data sheet for the MRF24J40 [68] states that a RSSI value of 0 represents a received signal power of -90 dBm or below while a RSSI value of 255 represents a received signal power of -35 dBm or above. Received power levels and corresponding RSSI values are tabulated in the data sheet but a corresponding power level is not provided for every RSSI value so interpolation had to be used to produce a table for converting RSSI values to power levels.

The base station contains a CC2591 range extender and a mini-circuits ZX10Q-2-27-S+ splitter between the RF transceiver and the antenna connections. The data sheet for the CC2591 states that it has a LNA gain of 11 dB [112] and the ZX10Q-2-27-S+ has a loss of about 3.4 dB in the 2.4 GHz band[5]. Therefore the power received by the RF transceiver will be approximately 8.6 dB higher than the power at the antenna terminals.

The data sheet for the MRF24J40 states that LQI values are based on the correlation degree between the spreading sequence and the incoming chips. This indicates how similar the incoming signal is to the expected signal. However the data sheet does not give any formula or table to relate between correlation degree and the LQI values. Instead it simply

[5] The actual values in the data sheet [73] are 3.45 dB for port 1 at 2.4 GHz, 3.37 dB for port 2 at 2.4 GHz, 3.40 dB for port 1 at 2.5 GHz and 3.44 dB for port 2 at 2.4 GHz

says that a LQI value of 0 represents a "very poor" link while a LQI value of 255 represents a "very good" link.

The RSSI for the nodes located near the base station is shown in Figures 9.38 to 9.43. Each point on the graph represents an individual measurement of RSSI. The values represent the power received by the RF transceiver in the base station after passing through the splitter and range extender. A RSSI value from the transceiver of 0 is shown on the graph as -90 dBm but the data sheet for the MRF24J40 states that it represents received signal power less than or equal to -90 dBm.

The author does not consider the RF transceiver to be a high quality measurement device and as such regards both the scaling and offset of the graph to be unreliable. Nevertheless the graphs do show how the signal loss varied across the season with much better temporal resolution than either the measurements with the propagation measurement system or the message loss rates.

A general pattern of greater distances leading to lower signal levels is seen but node 9 generally shows lower losses than node 8 despite node 8 being closer. As expected there is also a pattern over time with the lowest signal levels (and therefore highest propagation loss) being seen in the middle of the deployment with less loss at the beginning, when foliage was negligible and end, when the foliage had dried out.

RSSI results for nodes located near the repeater are shown in Figure 9.44. Again these values are in terms of the power received by the RF transceiver at the base station. Due to the technical issues with the repeater described in Section 9.1 usable data was only gathered early in the deployment. Interestingly the graph shows large jumps in signal level with the lower signal levels being seen more frequently at the start of the deployment. The author suspects that while these nodes were intended to be behind the repeater, they sometimes managed to associate directly with the base station, especially in the early stages of deployment when there was negligible foliage present.

The LQI graph for nodes directly connecting to the base station is shown in Figure 9.45. It shows a similar pattern of being worst in the middle of the season to the RSSI graph but the separation between nodes is much less pronounced. This is expected as signal level would only have a significant impact on correlation degree when the signal level is comparable to the noise level.

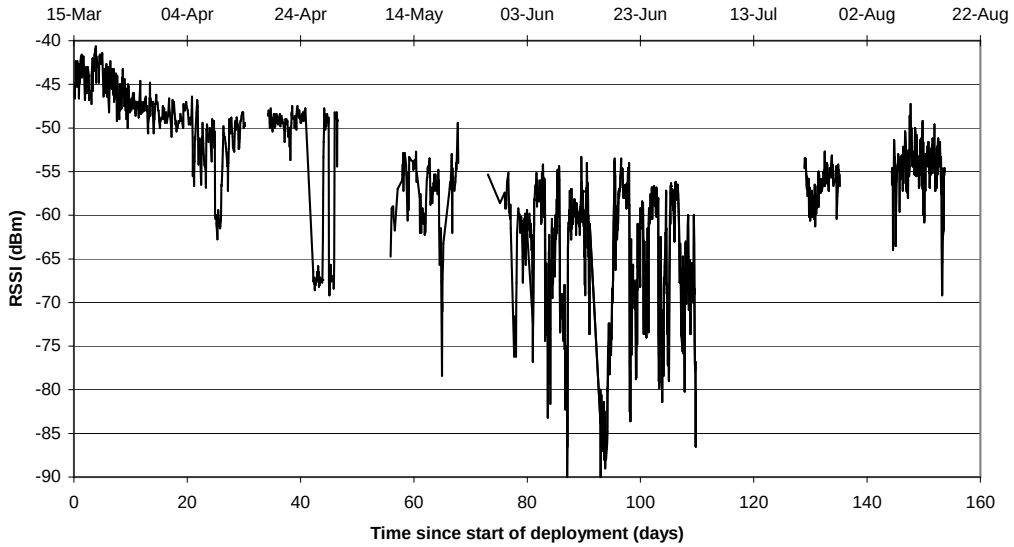


Figure 9.38: RSSI for node 7, 1 m from the base station.

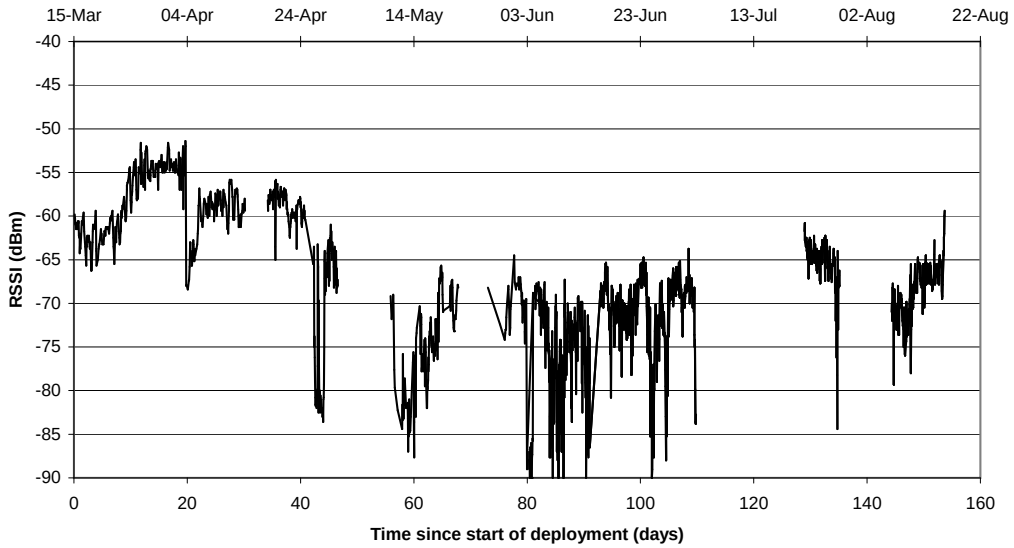


Figure 9.39: RSSI for node 8, 1.5 m from the base station.

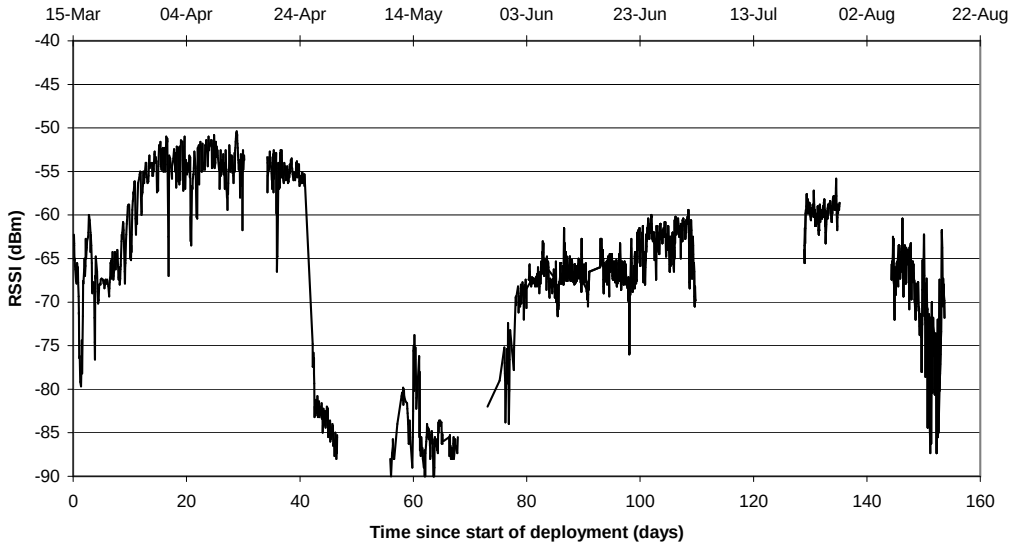


Figure 9.40: RSSI for node 9, 2 m from the base station.

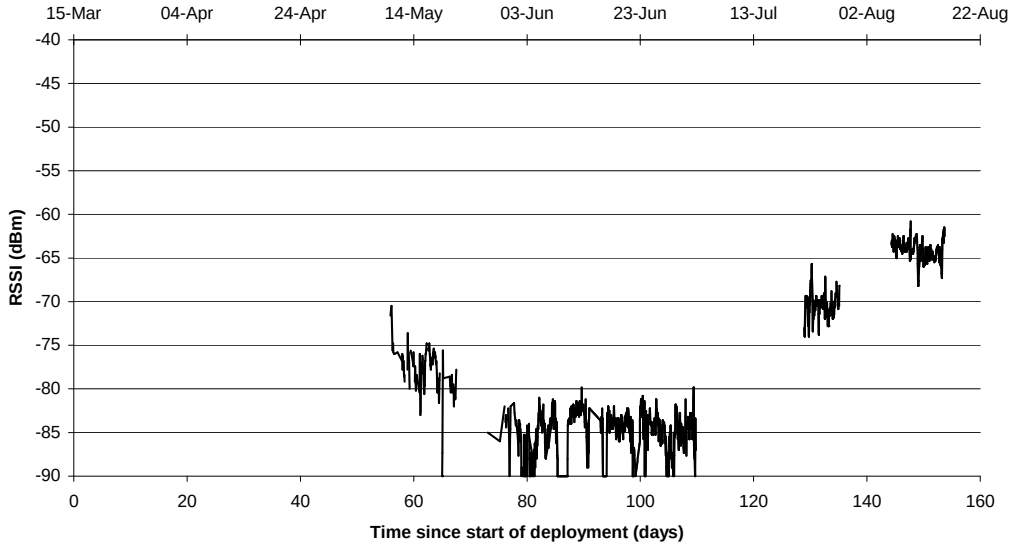


Figure 9.41: RSSI for node 13, 2.5 m from the base station.

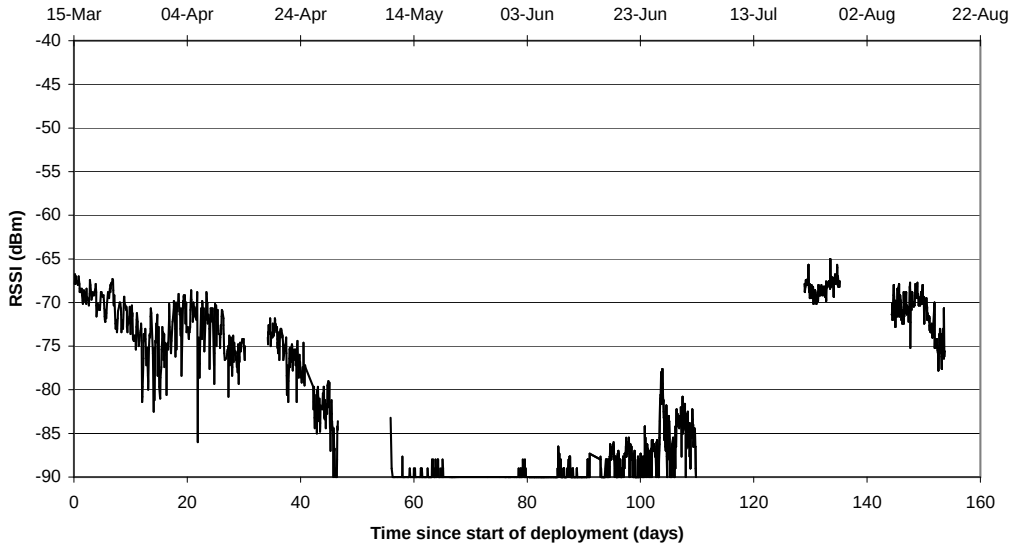


Figure 9.42: RSSI for node 11, 3 m from the base station.

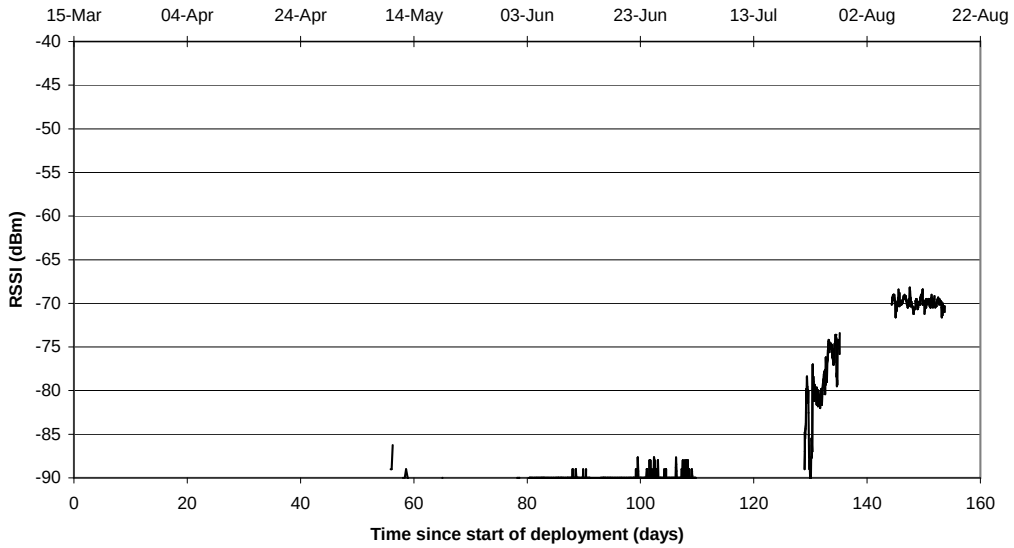


Figure 9.43: RSSI for node 14, 3.5 m from the base station.

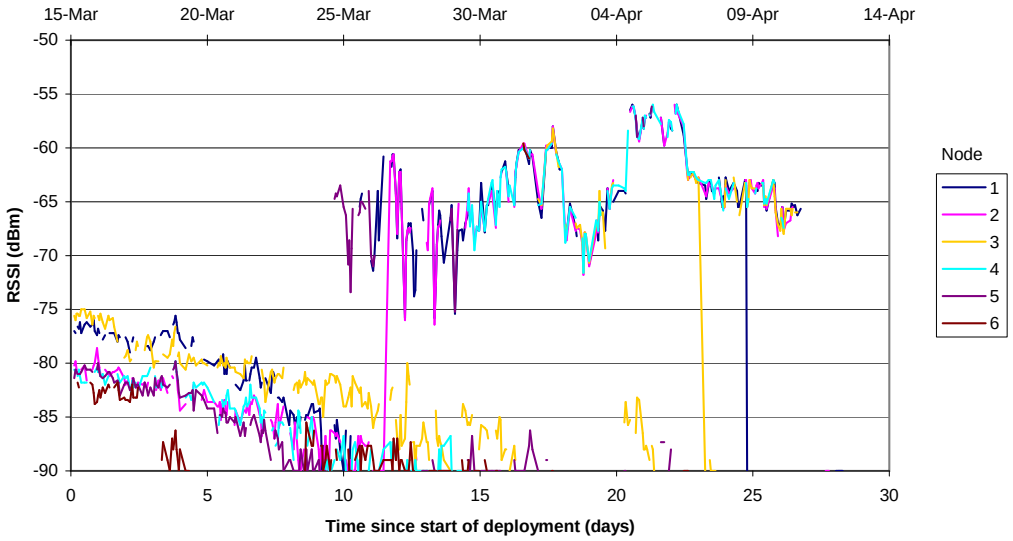


Figure 9.44: RSSI for nodes near repeater.

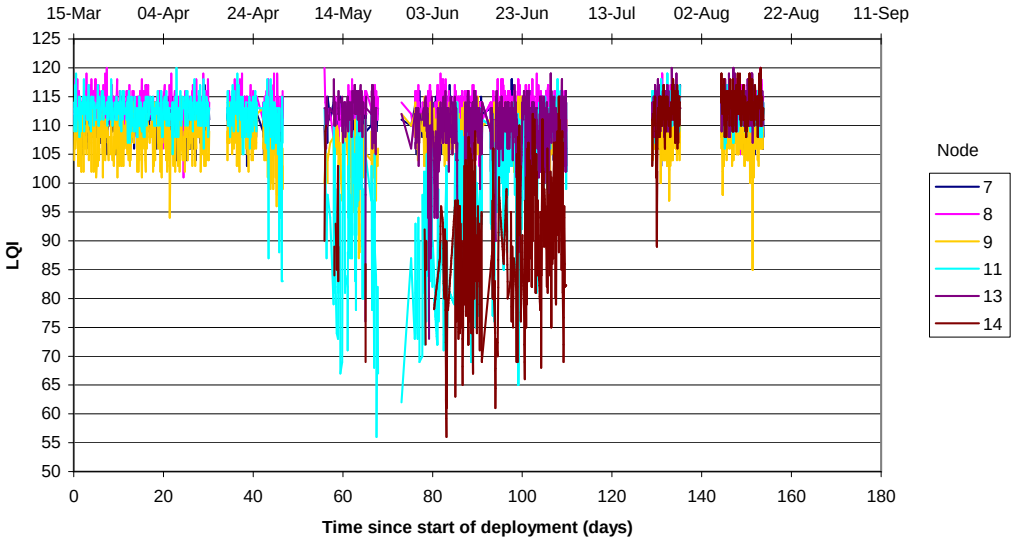


Figure 9.45: LQI for nodes near base station

9.3.4 Problems with clocks becoming unsynchronised

While attempting to investigate message loss rates it was noticed that the figures from the five day averages towards the end of the deployment appeared anomalous. In particular the three nodes behind the repeater that had reappeared were showing message loss rates of zero.

Investigation of the non-summarised results showed that starting with a reading at 19:10 on the 6th of August 2012, the message times were recorded as 10 past the hour rather than on the hour and hence were being ignored as being at unexpected times by the 5 day averaging process. This indicated that the clock used by the embedded Linux board in the base station was out of synchronization with the clock used by the sensor network hardware by at least five minutes. Investigating the raw data showed that whatever caused this was not a gradual drift but a sudden change. The cause is not clear.

To work around this and allow analysing the message loss figures for the whole deployment the rounding was changed from nearest 10 minutes to nearest 30 minutes.

9.4 Overall thoughts

The measurements show a large degree of variability. The author believes that to build a model of the radio signal loss within a rapeseed crop with antennas located close to ground level would require a large number of measurements taken over a large area. While measurements with antennas close to the ground were made in two locations during each session the locations were close together and close to the edge of the field. The experiments would also need to be carefully planned to minimise the impact of damage to the crop resulting from placing and removing the measurement equipment.

2.5 m was the largest distance over which communication was achieved with reasonable reliability throughout the deployment. It is suspected that improvements to the node design including use of a range extender in the end nodes and careful matching of antennas to the environment could improve this but given that the loss from the foliage was approximately 15 dB/m (based on Figure 8.3 and Table 9.1) the author suspects that the practical limit would be less than 5 m.

If the nodes in a wireless sensor network were arranged in a square grid with 5 m spacing there would be 400 nodes per hectare. Assuming that each node could be produced for £10 (an amount the author considers reasonable for large scale production) the nodes would cost £4000 per hectare. The author considers it unlikely that such a cost would be justifiable.

Chapter 10

Conclusions and suggestions for future work

This final chapter begins with a discussion of the equipment and techniques used. This is followed by a report of our findings on the propagation of radio waves in rapeseed, then a consideration of future work and finally ends by covering minor technical conclusions.

The power measurement system developed in Chapter 5 allowed the successful measurement of sensor node power consumption. Through a combination of measurements from the power measurement system and conservative battery capacity estimates the batteries in the end nodes were successfully made to last the whole season. No battery failures were observed with the repeater node either but the repeater node was out of action for a large proportion of the season due to water ingress problems. Power supply to the base station proved to be a larger problem and a source of significant downtime for the network as a whole.

The power measurement system could be improved in several ways, firstly the design error that led to a lower than planned frequency cut-off could be fixed. Secondly it would be useful to write software that could process and display the measurements in real time rather than relying on separate capture and post-processing. Thirdly it would be useful to remove the file size limitation so that longer periods of time could be captured.

The propagation measurement system was effective and allowed successful measurement of the radio loss through the rapeseed crop. However the system suffered from several issues. The first was that it took a long time to gather each measurement point limiting the number of points that could be gathered. The second was poor software design, the software saved the data to a fixed file name which had to be manually renamed after each run. This kept the software simple and was no problem in the laboratory but proved problematic in field work. The author found that laptop screens are very difficult to see when outdoors in sunny weather which made the renaming process difficult and error prone. It also led to inconsistent file naming which prevented the automating of the post processing and required each data file to be run though the processing software manually. This lead to the realisation that software to be used on laptops in the field needs to be designed to require a minimum of user interaction and to display important information in a large and bold manner.

Two designs of antenna stand were constructed for use in the propagation measurements near ground level. A design intended to be buried and provide a simulation of a sensor node and a flat plate intended to sit on the surface. The buried stand was quickly realised to be impractical to use due to the need to auger a relatively large hole in the ground for every antenna placement. The surface plate was very limited in where it could be placed within the crop making short distances difficult to accurately achieve. A stand with a small spike now seems like a better idea. It was also discovered that moving through fully grown rapeseed to place and remove measurement equipment damages the crop and while efforts were made to measure through undamaged crop, this was often difficult to achieve. One way to solve this may be to build a scaffold over the crop and place measurement gear by reaching down vertically but such a system would require considerable effort to install and remove.

Sensor nodes were deployed at Tatton Dale Farm in March 2012 along with a base station and repeater. Two groups of nodes were deployed one group transmitting directly to the base station and the other transmitting via the repeater node. Unfortunately the repeater was out of action for much of the season due to technical problems related to water ingress. The group transmitting directly to the base station operated successfully for most of the season, though with substantial gaps in the data caused by the base station runnning out of power.

Both the measurements from the propagation measurement system and the results from the deployment showed extreme signal loss when transmitting through fully grown rapeseed. Furthermore the signal loss was highly variable. Variation was seen in several forms, firstly there was a large amount of variation with frequency within individual measurement runs from the propagation measurement system. Secondly there was large variation between measurement runs through different sections of crop at the same distance. Finally large fluctuations were seen in the received signal strength indication values returned by the sensor nodes in the deployment.

As would be expected the propagation loss also varied massively through the season, being very low initially when the plant matter was close to nonexistant. Peaking when the rapeseed was fully grown and then reducing again as the crop dried out at the end of the season.

The author believes that the measurements of propagation loss and message loss taken during this project were sufficient to establish that transmitting through fully grown rapeseed near ground level at 2.4 GHz is limited to distances of a few metres. The author strongly feels that nodes at such a close spacing are unlikely to be affordable in a commerical agricultural deployment. However such transmission could potentially be useful if readings from a cluster of closely spaced readings were desired to provide extremely localised information during a crop trial.

Propagation loss near ground level in rapeseed at 2.4 GHz was found to be highly variable and therefore to build a model of the propagation loss in rapeseed would require a much larger number of measurements so that increases in loss with distance could be separated from other variations. Investigating other frequency ranges may be useful but it is felt unlikely that they would give sufficient improvement to achieve distances that would support the low node densities needed to achieve a viable deployment on a farm. Operating just below the

crop canopy gives better results than transmitting near ground level but significantly worse results than operating just above the crop canopy. Since the crop canopy changes in height over the season the author does not believe that operating just below the crop canopy would offer any compelling advantage over operating just above it. The author noticed no apparent difference between transmitting above the crop canopy and transmitting above open ground but such measurements were not a focus of the project and only a small number of them were made.

Transmitting above the crop canopy appears to be the only reasonable option. As such it would be useful to have an antenna system that can operate within this area without interfering with any crop spraying operations. The author has been advised that spray booms are normally operated at just above the current height of the crop. Of course this creates a problem, an antenna installed to be just above the crop canopy when the crop is fully grown will be massively above it earlier in the season. This means it must either be able to be adjusted in height as the season progresses (which is difficult because of the difficulties performing practical work within a rapeseed crop) or be able to tolerate being hit by a spray boom without damaging either the spray boom or the antenna. The author initially thought this could be achieved by having an antenna stand with a sprung pivot at the bottom so it could be knocked over and then spring back up. However it is now felt that this is unlikely to work due to the tangled nature of rapeseed and that to survive being knocked over an antenna system would have to be able to bend in multiple places.

The author believes that to determine if it is possible to design and construct a successful and economical wireless sensor network for operation within a rapeseed crop and to design and construct it if it was determined to be possible would require a multidisciplinary approach. Input would be required from agricultural researchers on what needs to be measured, how accurate those measurements need to be, what spacial and temporal resolution would be desirable and what the benefits would be. Input would be required from electronics manufacturing experts on how best to design the nodes and how much they would cost to mass produce. Input would be required from farmers on how to place the nodes and their antennas so they provided representative information on the conditions in the crop while minimising interference with farming practices. Careful consideration of routing algorithms would be required including consultation with experts in this field. It is likely that even after extensive consultations that there would still be unanswered questions which would have to be resolved through further experimental work. In particular the author feels it is likely that trials would be needed to answer the question of just how much benefit could be expected by the farmer from operating the network.

Propagation loss in and around rapeseed could be investigated under many different conditions. The author would suggest investigating other frequency bands, distances greater than 30 m and asymmetric combinations of antenna height. In particular the author feels it would be worth investigating whether a single antenna placed high above the field could collect data over long distances from a number of nodes with antennas located just above the crop canopy.

In addtion to the knowledge gained about radio propagation loss in rapeseed the author came to a number of minor conclusions which were felt to be sufficiently interesting to future experimenters to mention here.

It became clear over the course of the project that actually performing any work in a rapeseed crop is difficult. The plants tangle together such that attempting to walk through the crop requires considerable effort to separate the tangled plants and inevitably causes damage to the crop. This means any maintenance work on a trial network is difficult and risks affecting the accuracy of results. The same problem applies to measurements with test equipment as placing and removing the test antennas inevitably causes crop damage. Finally it means that any practical final deployment must be maintenance-free being able to be placed at the start of a growing season and left in place until the end of that growing season. It also raises practical concerns for removing the nodes before harvest.

The size of a farm field really only became apparent when activities moved to Tatton Dale Farm. The small size of the field used at Rothamsted had seriously distorted the author's perceptions. As a result the author did not realise how impractical node spacings of only a couple of metres would be until it was too late to change the deployment plan significantly. The author has also realised that due to the large size of a farm simple one-hop and two-hop networks would not be practical and that routing protocols capable of handling multiple hops would be required.

Burying nodes seemed like a good idea early on in the project. However the author now believes that while it produces a neat looking deployment it is in fact largely pointless. The dense crop means that an above ground node enclosure would be shielded from damage for most of the growing season and an above ground location is far less likely to suffer from water ingress problems if the enclosure is imperfectly sealed.

The MiWi software stack was chosen by the author due to familiarity with Microchip products. However it did not prove to be greatly suited to wireless sensor network work. It had to be modified by the author to allow repeater nodes to sleep. It requires a significant number of packets to be exchanged for a new node to join the network. It does not provide any means to track RSSI across multiple hops (even retrieving RSSI from the final hop required modifications) and only allows a maximum of two hops between the base station and any end node.

The author has learnt that great care is needed when reading specifications and data sheets for components and equipment. In particular the conditions under which those specifications are valid must be carefully compared to the conditions under which the components and equipment will actually be used. It is also necessary to be careful about what is omitted in the data sheets, it is common for manufacturers to give typical values but not to give minimums and maximums for those values.

The author has learnt, both from experiences with the project and experience outside the project that while mobile Internet services appear initially to be a convenient and economical way to transport the data from a wireless sensor network, that in practice reliability of such services in the countryside is poor. SMS text messages are far more reliable but also more expensive. In some cases it may also be possible to avoid the use of the cellular network completely and either extend the wireless sensor network to take the data to where it is needed (for example a farm office) or use another private wireless network (for example a Wi-Fi network).

The author believes there would be value in having a laboratory test environment in which

a large number of nodes could be placed in screened containers and connected through attenuators and splitters to test the interaction of a large number of nodes with propagation losses comparable to those likely to be encountered in the field. This would allow more effective testing prior to deployment.

References

[1] ADAS. Evaluation of the impact on UK agriculture of the proposal for a regulation of the European Parliament and of the council concerning the placing of plant protection products on the market, 2008. http://www.adas.co.uk/LinkClick.aspx?fileticket=jeVQ5K1z3iQ%3D&tabid=246

[2] Agilent Technologies. *Agilent N6780 Series Source/Measure Units (SMUs) for the N6700 Modular Power System*, 2010. http://cp.literature.agilent.com/litweb/pdf/5990-5829EN.pdf

[3] Agilent Technologies. Battery Drain Video: Seamless Measurement Ranging on the N6781A, 2010. http://www.home.agilent.com/agilent/editorial.jspx?cc=GB&lc=eng&ckey=1917598&nid=-35714.937221.02&id=1917598&cmpid=MC9667

[4] Agilent Technologies. *E4428C/38C ESG Signal Generators User's Guide*, 2010. http://cp.literature.agilent.com/litweb/pdf/E4400-90503.pdf

[5] J.N. Al-Karaki and A.E. Kamal. Routing techniques in wireless sensor networks: a survey. *Wireless Communications, IEEE*, 11(6):6–28, 2004. ISSN 1536-1284.

[6] R. Alena, D. Evenson, and V. Rundquist. Analysis and testing of mobile wireless networks. In *Aerospace Conference Proceedings, 2002. IEEE*, volume 3, pages 3–1131 – 3–1144 vol.3, 2002. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1035243

[7] Analog Devices. *AD8397 Rail-to-Rail, High Output Current Amplifier*, 2005. http://www.analog.com/static/imported-files/data_sheets/AD8397.pdf

[8] Analog Devices. AD524 PrecisionInstrumentation Amplifier, 2007. http://www.farnell.com/datasheets/233358.pdf

[9] Analog Devices. *ADuM4160 Full/Low Speed 5 kV USB Digital Isolator*, 2010. http://www.analog.com/static/imported-files/data_sheets/ADuM4160.pdf

[10] G. Anastasi, A. Falchi, A. Passarella, M. Conti, and E. Gregori. Performance measurements of motes sensor networks. In *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, MSWiM '04, pages 174–181, New York, NY, USA, 2004. ACM. ISBN 1-58113-953-5. http://doi.acm.org/10.1145/1023663.1023695

[11] Anritsu. *VNA Master MS2024A/26A/34A/36A User Guide*, 2008. http://www.anritsu.com/en-US/Downloads/Manuals/User-Guide/DWL2733.aspx

[12] Hermann Auernhammer. Precision farming – the environmental challenge. *Computers and Electronics in Agriculture*, 30(1-3):31 – 43, 2001. ISSN 0168-1699. http://www.sciencedirect.com/science/article/B6T5M-429Y516-4/2/f3a80f6f54f3cf73485e2cd09d3c4bd3

[13] Australian Centre for International Agricultural Research. Reducing water use in agriculture, 2006. http://aciar.gov.au/files/node/725/Reducing%20water%20use%20in%20agriculture%20layout.pdf

[14] Aline Baggio. Wireless sensor networks in precision agriculture. In *First REALWSN 2005 Workshop on Real-World Wireless Sensor Networks*, 2005. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.120.46&rep=rep1&type=pdf

[15] G Baldauf. Removal of Pesticides in Drinking Water Treatment. *Acta hydrochimica et hydrobiologica*, 21(4):203–208, 1993. http://onlinelibrary.wiley.com/doi/10.1002/aheh.19930210403/pdf

[16] BeagleBoard.org. *BeagleBoard SystemReference Manual Revision C4*, 2009. http://beagle.s3.amazonaws.com/design/Beagle_SRM_C4_0_0.pdf

[17] Richard Beckwith, Dan Teibel, and Pat Bowen. Report from the Field: Results from an Agricultural Wireless Sensor Network. *Local Computer Networks, Annual IEEE Conference on*, 0:471–478, 2004. ISSN 0742-1303. http://dx.doi.org/10.1109/LCN.2004.105

[18] E. F. Boys, S. E. Roques, J. S. West, C. P. Werner, G. J. King, P. S. Dyer, and B. D. L. Fitt. Effects of R gene-mediated resistance in Brassica napus (oilseed rape) on asexual and sexual sporulation of Pyrenopeziza brassicae (light leaf spot). *Plant Pathology*, 61(3):543–554, 2012. ISSN 1365-3059. http://dx.doi.org/10.1111/j.1365-3059.2011.02529.x

[19] Michael Buettner, Gary V Yee, Eric Anderson, and Richard Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 307–320. ACM, 2006. http://dl.acm.org/citation.cfm?id=1182838

[20] Chiara Buratti, Andrea Conti, Davide Dardari, and Roberto Verdone. An Overview on Wireless Sensor Networks Technology and Evolution. *Sensors*, 9(9):6869–6896, 2009. ISSN 1424-8220. http://www.mdpi.com/1424-8220/9/9/6869

[21] M.J Cerejeira, P Viana, S Batista, T Pereira, E Silva, M.J Valerio, A Silva, M Ferreira, and A.M Silva-Fernandes. Pesticides in Portuguese surface and ground waters. *Water Research*, 37(5):1055 – 1063, 2003. ISSN 0043-1354. http://www.sciencedirect.com/science/article/pii/S0043135401004626

[22] Rohana Chandrajith, Shirani Seneviratna, Kumudu Wickramaarachchi, T. Attanayake, T.N.C. Aturaliya, and C.B. Dissanayake. Natural radionuclides and trace elements in rice field soils in relation to fertilizer application: study of a chronic kidney disease area in Sri Lanka. *Environmental Earth Sciences*, 60(1):193–201, 2010. ISSN 1866-6280. http://dx.doi.org/10.1007/s12665-009-0179-1

[23] COMMISSION OF THE EUROPEAN COMMUNITIES. A Thematic Strategy on the Sustainable Use of Pesticides, 2006. http://ec.europa.eu/environment/ppps/pdf/com_2006_0372.pdf

[24] P. Corke, T. Wark, R. Jurdak, Wen Hu, P. Valencia, and D. Moore. Environmental Wireless Sensor Networks. *Proceedings of the IEEE*, 98(11):1903 –1917, nov. 2010. ISSN 0018-9219. http://dx.doi.org/10.1109/JPROC.2010.2068530

[25] Roger Coude. Radio Mobile WEB Site. http://www.cplus.org/rmw/english1.html

[26] Crossbow Technology. *MICA2 WIRELESS MEASUREMENT SYSTEM*, 2005. http://www.sim.uni-hannover.de/~svs/theses/Mica2Datasheet.pdf

[27] Dominic J.P. Crutchley. *Wireless sensor networks in hostile RF environments.* PhD thesis, University of Manchester, 2012. https://www.escholar.manchester.ac.uk/uk-ac-man-scw:167409

[28] Data Translation. *DT9816 User's Manual*, 2010. ftp://ftp.datx.com/public/dataacq/dt9816/manuals/DT9816UM.pdf

[29] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '08: ACM SIGGRAPH 2008 classes*, pages 1–10, New York, NY, USA, 2008. ACM. http://doi.acm.org/10.1145/1401132.1401174

[30] Department for Environment, Food and Rural Affairs. Agriculture in the United Kingdom 2011, 2012. http://www.defra.gov.uk/statistics/files/defra-stats-foodfarm-crosscutting-auk-auk2011-120709.pdf

[31] John Dunlop and Joan Cortes. Impact of Directional Antennas in Wireless Sensor Networks. *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference*, 0:1–6, 2007. http://dx.doi.org/10.1109/MOBHOC.2007.4428717

[32] Duracell. *MN1500 datasheet Issue 8A*, 2006. http://www.rapidonline.com/pdf/18-3260e.pdf

[33] D. Ehlert, J. Schmerler, and U. Voelker. Variable Rate Nitrogen Fertilisation of Winter Wheat Based on a Crop Density Sensor. *Precision Agriculture*, 5(3):263–273, 2004. ISSN 1385-2256. http://dx.doi.org/10.1023/B%3APRAG.0000032765.29172.ec

[34] eLinux.org. BeagleBoard Zippy. http://elinux.org/BeagleBoard_Zippy

[35] Robert Stratman Elliott. *Antenna theory and design.* Prentice-Hall inc, 1981.

[36] Bob Evans. Pesticide run off into English rivers – a big problem for farmers. *Pesticides News*, September 2009. http://www.pan-uk.org/pestnews/Free%20Articles/PN85_12-15.pdf

[37] Andy Floyd. Measurement and analysis of alkaline battery performance for low power wireless applications. Master's thesis, The University of Manchester, 2011. Draft A [Review 150811].

[38] Charles Samuel Franklin. Improvements in Wireless Telegraph and telephone Aerials, 1924. http://www.aktuellum.com/circuits/antenna-patent/patents/242342.pdf

[39] Vincent F Fusco. *Foundations of antenna theory and techniques*. Pearson, 2005.

[40] Patrick Gaydecki. *Foundations of digital signal processing: theory, algorithms and hardware design*. The institution of Electrical Engineers, 2004.

[41] D Goense, John Thelen, and K Langendoen. Wireless sensor networks for precise Phytophthora decision support. In *5th European Conference on Precision Agriculture (5ECPA), Uppsala, Sweden*, 2005. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.78.5785&rep=rep1&type=pdf

[42] Martin A. Hebel, Ralph Tate, and Dennis G. Watson. Results of Wireless Sensor Network Transceiver Testing for Agricultural Applications. In *2007 ASABE Annual International Meeting*, 2007. http://elibrary.asabe.org/azdez.asp?JID=5&AID=23186&t=2&v=&i=&CID=min2007&redir=&redirType=&downPDF=Y

[43] Martin A. Hebel, Ralph Tate, and Dennis G. Watson. WSN Link Budget Analysis for Precision Agriculture. In *2008 ASABE Annual International Meeting*, 2008. https://elibrary.asabe.org/azdez.asp?JID=5&AID=24935&t=2&v=&i=&CID=prov2008&redir=&redirType=&downPDF=Y

[44] R.J. Hillocks. Farming with fewer pesticides: {EU} pesticide review and resulting challenges for {UK} agriculture. *Crop Protection*, 31(1):85 – 93, 2012. ISSN 0261-2194. http://www.sciencedirect.com/science/article/pii/S026121941100264X

[45] Christof Hubner, Rachel Cardell-Oliver, Rolf Becker, Klaus Spohrer, Kai Jotter, and Tino Wagenknecht. Wireless soil moisture sensor networks for environmental monitoring and vineyard irrigation. In *Proceedings of the 8th International Conference on Electromagnetic Wave Interaction with Water and Moist Substances (ISEMA 2009)*, 2009.

[46] IEEE Computer Society. IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs), 2006. http://dx.doi.org/10.1109/IEEESTD.2006.232110

[47] ITT. *Reference data for radio engineers*. Howard W. Sams and Co. INC, sixth edition, 1975. ISBN 0-672-21218-8.

[48] ITU Radiocommunication Assembly. RECOMMENDATION ITU-R P.833-2 ATTENUATION IN VEGETATION. Technical report, 1999. http://www.catr.cn/radar/itur/201007/P020100714465224514150.pdf

[49] John F. Janek and Jeffrey J. Evans. Predicting Ground Effects of Omnidirectional Antennas in Wireless Sensor Networks. *Wireless Sensor Network*, 2:879–890, 2010. http://dx.doi.org/10.4236/wsn.2010.212106

[50] Yulong Jin and Jiaqiang Yang. Design an Intelligent Environment Control System for GreenHouse Based on RS485 Bus. *2012 Third International Conference on Digital Manufacturing & Automation*, 0:361–364, 2011. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6052026

[51] T.J. Judasz and B.B. Balsley. Improved theoretical and experimental models for the coaxial colinear antenna. *Antennas and Propagation, IEEE Transactions on*, 37(3): 289–296, 1989. ISSN 0018-926X.

[52] R. Jurdak, P. Baldi, and C.V. Lopes. Adaptive Low Power Listening for Wireless Sensor Networks. *Mobile Computing, IEEE Transactions on*, 6(8):988–1004, 2007. ISSN 1536-1233.

[53] Peterpaul Klein Haneveld. Evading Murphy: A Sensor Network Deployment in Precision Agriculture, 2007. http://www.st.ewi.tudelft.nl/~koen/papers/LOFAR-agro-take2.pdf

[54] Niina Kotamaki, Sirpa Thessler, Jari Koskiaho, Asko O. Hannukkala, Hanna Huitu, Timo Huttula, Jukka Havento, and Markku Jarvenpaa. Wireless in-situ Sensor Network for Agriculture and Water Monitoring on a River Basin Scale in Southern Finland: Evaluation from a Data User's Perspective. *Sensors*, 9(4):2862–2883, 2009. ISSN 1424-8220. http://www.mdpi.com/1424-8220/9/4/2862

[55] Maxim Krasnyansky. Virtual tunnel. http://vtun.sourceforge.net/

[56] R.J. Lamb. Entomology of oilseed Brassica crops. *Annual Review of Entomology*, 34(1):211–229, 1989. http://www.annualreviews.org/doi/pdf/10.1146/annurev.en.34.010189.001235

[57] K.G. Langendoen, A. Baggio, and O.W. Visser. Murphy Loves Potatoes: Experiences from a Pilot Sensor Network Deployment in Precision Agriculture. In *14th Int. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, pages 1–8, April 2006. ISBN 1-4244-0054-6. http://www.st.ewi.tudelft.nl/~koen/papers/WPDRTS06.pdf

[58] J.D. Lea-Cox, F. Arguedas Rodriguez, A.G. Ristvey, D.S. Ross, and G Kantor. Wireless Sensor Networks to Precisely Monitor Substrate Moisture and Electrical Conductivity Dynamics in a Cut-Flower Greenhouse Operation. Acta Hort. (ISHS), 2011. http://sensornet.umd.edu/publications/Lea-Cox%20et%20al.%202011.%20Acta%20Hort.%20893_1057-1063.pdf

[59] John D. Lea-Cox, Stephen Black, Andrew Ristvey, and David S. Ross. Towards Precision Scheduling of Water and Nutrient Applications, Utilizing a Wireless Sensor Network on an Ornamental Tree Farm. In *SNA Research Conference*, 2008. http://smart-farms.net/sites/default/files/publications/Lea-Cox%20et%20al,%202008%20SNA%2053_32-37.pdf

[60] John D. Lea-Cox, Andrew G. Ristvey, Felix Arguedas Rodriguez, David S. Ross, Joshua Anhalt, and George Kantor. A Low-cost Multihop Wireless Sensor Network, Enabling Real-Time Management of Environmental Data

for the Greenhouse and Nursery Industry. In *International Symposium on High Technology for Greenhouse System Management: Greensys2007*, 2008. http://sensornet.umd.edu/Publications/Lea-Cox%20et%20al,%202008%20Acta%20Hort%20Sensor%20Networks%20Pre-Print.pdf

[61] Thomas H. Lee. *Planar Microwave Engineering*. Cambridge University Press, first edition, 2004. ISBN 0-521-83526-7.

[62] Hui Liu, Zhijun Meng, and Yuanyuan Shang. Sensor Nodes Placement for Farmland Environmental Monitoring Applications. In *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on*, pages 1–4, 2009.

[63] Lofar agro. http://goo.gl/shfn4

[64] Nick Long. Frequency bands for short range devices. In *The IEE Seminar on Telemetry and Telematics, 2005*, pages 6/1–6/5, 2005. http://dx.doi.org/10.1049/ic:20050103

[65] Maxim Integrated Products. *MAX16904 2.1MHz, High-Voltage, 600mA Mini-Buck Converter*, 2011. http://datasheets.maxim-ic.com/en/ds/MAX16904.pdf

[66] Pedro Mestre, Jose Ribeiro, Carlos Serodio, and Joao Monteriro. Propagation of IEEE802.15.4 in Vegetation. In *Proceedings of the World Congress on Engineering 2011 Vol II*, 2011. http://www.iaeng.org/publication/WCE2011/WCE2011_pp1786-1791.pdf

[67] Microchip Technology. MiWi Wireless Networking Protocol Stack, 2007. http://ww1.microchip.com/downloads/en/appnotes/01066a.pdf

[68] Microchip Technology Inc. *MRF24J40 Data Sheet*, 2008. http://ww1.microchip.com/downloads/en/DeviceDoc/DS-39776b.pdf

[69] Microchip Technology Inc. *PICDEM Z Demonstration Kit User's Guide*, 2008. http://ww1.microchip.com/downloads/en/AppNotes/PICDEM%20Z%20Users_Guide_51524C.pdf

[70] Microchip Technology Inc. *MCP9700/9700A, MCP9701/9701A Low-Power Linear Active Thermistor ICs*, 2009. http://ww1.microchip.com/downloads/en/DeviceDoc/21942e.pdf

[71] Microchip Technology Inc. MRF24J40MC 2.4 GHz IEEE Std. 802.15.4 RF Transceiver Module with PA/LNA and External Antenna Connector, 2011. http://ww1.microchip.com/downloads/en/DeviceDoc/75002A.pdf

[72] Microchip Technology Inc. *PIC18F46J11 Family Data Sheet*, 2011. http://ww1.microchip.com/downloads/en/DeviceDoc/39932D.pdf

[73] Mini-Circuits. *ZX10Q-2-27 SMA Connectorized Power Splitter/Combiner*, 2012. http://www.minicircuits.com/pdfs/ZX10Q-2-27.pdf

[74] Andreas F. Molisch. *Wireless Communications*. John Wiley and Sons Ltd, 2005.

[75] Miguel Morales. Wireless Sensor Monitor Using the eZ430-RF2500. Technical report, Texas Instruments, 2011. http://www.ti.com/lit/an/slaa378d/slaa378d.pdf

[76] Murata Power Solutions. UEI15 Series Isolated Wide Input Range 15-Watt DC/DC Converters, 2013. http://docs-europe.electrocomponents.com/webdocs/0ce7/0900766b80ce7679.pdf

[77] National Instruments. *NI-VISA User Manual*, 2001. http://www.ni.com/pdf/manuals/370423a.pdf

[78] G. Ngandu, C. Nomatungulula, S. Rimer, B.S. Paul, K. Ouahada, and B. Twala. Evaluating effect of foliage on link reliability of wireless signal. In *Industrial Technology (ICIT), 2013 IEEE International Conference on*, pages 1528–1533, 2013.

[79] Martin Nilsson. Directional antennas for wireless sensor networks. In *Proceedings of 9th Scandinavian Workshop on Wireless Adhoc Networks (Adhoc'09)*, page 4, Uppsala, Sweden, 2009. http://eprints.sics.se/3605/01/adhoc2009_final_paper.pdf

[80] NXP. ZigBee PRO Stack User Guide, 2012. http://www.nxp.com/documents/user_manual/JN-UG-3048.pdf

[81] Brendan O'Flynn, D. Laffey, J. Buckley, J. Barton, and S.C. O'Mathuna. Simulation, design, development and test of antennas for wireless sensor network systems. *Microelectronics International*, 24(2):3–6, 2007. http://www.emeraldinsight.com/journals.htm?articleid=1602727&show=abstract

[82] Susan A. O'Shaughnessy and Steven R. Evett. Integration of Wireless Sensor Networks into Moving Irrigation Systems for Automatic Irrigation Scheduling. In *2008 ASABE Annual International Meeting*, 2008. http://www.researchgate.net/publication/237293592_Integration_of_Wireless_Sensor_Networks_into_Moving_Irrigation_Systems_for_Automatic_Irrigation_Scheduling

[83] Meng-Shiuan Pan, Hua-Wei Fang, Yung-Chih Liu, and Yu-Chee Tseng. Address assignment and routing schemes for ZigBee-based long-thin wireless sensor networks. In *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 173–177. IEEE, 2008. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4525604&tag=1

[84] J.D. Parsons. *The Mobile Radio Propagation Channel*. John Whiley & Sons LTD, second edition, 2000. ISBN 0-471-98857-X.

[85] M. Polivka and A. Holub. Collinear and Coparallel Principles in Antenna Design. In *Progress In Electromagnetics Research Symposium*, 2007. http://piers.org/piersproceedings/download.php?file=cGllcnMyMDA3UHJhZ3VlfDNBMmJfMDMzNy5wZGZ8MDcwMjIwMTgzNzQ4

[86] Sandra Postel. Sustaining Freshwater and Its Dependents. In *State of the World 2013*, pages 51–62. Island Press/Center for Resource Economics, 2013. ISBN 978-1-59726-415-0. http://dx.doi.org/10.5822/978-1-61091-458-1_5

[87] Martin Pot. Anatomy of a 2.4GHz Rubber Ducky Antenna, 2005. http://martybugs.net/wireless/rubberducky.cgi

[88] Martin Pot. Home-brew Compact 6dBi Collinear Antenna, 2005. http://martybugs.net/wireless/collinear.cgi

[89] QinetiQ. A Generic Model of 1-60 GHz Radio Propagation through Vegetation - Final Report, 2002. http://www.ofcom.org.uk/static/archive/ra/topics/research/topics/propagation/vegetation/vegetation-finalreportv1_0.pdf

[90] A. Rahman, W. Olesinski, and P. Gburzynski. Controlled flooding in wireless ad-hoc networks. In *Wireless Ad-Hoc Networks, 2004 International Workshop on*, pages 73–78, 2004. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1525544

[91] T.R. Rao and D. Balachander. RF propagation experiments in agricultural fields and gardens for wireless sensor communications. *Progress In Electromagnetics Research*, 2013. http://www.jpier.org/PIERC/pier.php?paper=13030710

[92] T.R. Rao, D. Balachander, N. Tiwari, and P. Mvsn. Ultra-high frequency near-ground short-range propagation measurements in forest and plantation environments for wireless sensor networks. *Wireless Sensor Systems, IET*, 3(1):80–84, 2013. ISSN 2043-6386.

[93] Rohde & Schwarz. *R&S FSH4/8 Handheld Spectrum Analyzer Operating Manual*, 2010. http://www.rohde-schwarz.co.uk/file_11191/FSH_Operating_Manual_en_v1.50.pdf

[94] Rothamsted. Guide to the Classical and other Long-term Experiments, Datasets and Sample Archive. Technical report, 2006. http://www.rothamsted.ac.uk/resources/LongTermExperiments.pdf

[95] Rothamsted Research. http://www.rothamsted.bbsrc.ac.uk/

[96] Luis Ruiz-Garcia, Loredana Lunadei, Pilar Barreiro, and Ignacio Robla. A Review of Wireless Sensor Technologies and Applications in Agriculture and Food Industry: State of the Art and Current Trends. *Sensors*, 9(6):4728–4750, 2009. ISSN 1424-8220. http://www.mdpi.com/1424-8220/9/6/4728

[97] S. Salous, N. Nikandrou, and N.F. Bajj. Digital techniques for mobile radio chirp sounders. *Communications, IEE Proceedings-*, 145(3):191 –196, jun 1998. ISSN 1350-2425. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=689416

[98] SANYO. New eneloop Batteries Remains Capacity Longer, 2011. http://panasonic.net/sanyo/news/2011/10/06-1.pdf

[99] Thomas Schwengler. Radio Propagation Modeling, 2013. http://morse.colorado.edu/~tlen5510/text/classwebch3.html

[100] Andrew Seville. Effects of Propagation Between the Indoor and Outdoor Environment Report on initial trials D 50-1, 2001. http://www.ofcom.org.uk/static/archive/ra/topics/research/rcru/project50/D50-1.pdf

[101] C.E. Shannon. Communication In The Presence Of Noise. *Proceedings of the IEEE*, 86(2):447 –457, feb. 1998. ISSN 0018-9219. http://http://dx.doi.org/10.1109/JPROC.1998.659497

[102] Signal Wizard Systems. http://www.signalwizardsystems.com/

[103] R. Skaug and J.F. Hjelmstad. *Spread spectrum in communication*. Peter Peregrinus Ltd, first edition, 1985. ISBN 0-863-41034-0.

[104] Val H. Smith and David W. Schindler. Eutrophication science: where do we go from here? *Trends in Ecology & Evolution*, 24(4):201 – 207, 2009. ISSN 0169-5347. http://www.sciencedirect.com/science/article/pii/S016953470900041X

[105] K. Solbach. Microstrip-Franklin Antenna. *Antennas and Propagation, IEEE Transactions on*, 30(4):773–775, 1982. ISSN 0018-926X.

[106] ST Microelectronics. TL071 Low noise JFET single operational amplifier, 2008. http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/DATASHEET/CD00000488.pdf

[107] David R Steward, Paul J Bruss, Xiaoying Yang, Scott A Staggenborg, Stephen M Welch, and Michael D Apley. Tapping unsustainable groundwater stores for agricultural production in the High Plains Aquifer of Kansas, projections to 2110. *Proceedings of the National Academy of Sciences*, 110(37):E3477–E3486, 2013.

[108] R. Dean Straw. *The ARRL Antenna Book*. ARRL, 21st edition, 2007.

[109] Pablo Suarez, Carl-Gustav Renmarker, Adam Dunkels, and Thiemo Voigt. Increasing ZigBee network lifetime with X-MAC. In *Proceedings of the workshop on Real-world wireless sensor networks*, REALWSN '08, pages 26–30, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-123-1. http://doi.acm.org/10.1145/1435473.1435481

[110] Texas instruments. *CC24200 RSSI*, 2008. http://www.ti.com/lit/ds/swrs040c/swrs040c.pdf

[111] Texas instruments. *CC2500 Low-Cost Low-Power 2.4 GHz RF Transceiver*, 2008. http://www.ti.com/lit/ds/swrs040c/swrs040c.pdf

[112] Texas Instruments. *CC2591 2.4-GHz RF Front End*, 2008. http://focus.ti.com/lit/ds/symlink/cc2591.pdf

[113] Texas Instruments. *eZ430-RF2500 Development Tool*, 2009. http://focus.ti.com/lit/ug/slau227e/slau227e.pdf

[114] Texas instruments. *MSP430F22x2 MSP430F22x4 MIXED SIGNAL MICROCONTROLLER*, 2012. http://www.ti.com/lit/ds/slas504g/slas504g.pdf

[115] John Thelen and Daan Goense. Radio wave propagation in potato fields. In *1st Workshop on Wireless Network Measurements*, 2005. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.74.6308&rep=rep1&type=pdf

[116] Thurlby Thandar Instruments. *TG1010 10MHz D.D.S. Function Generator*, 1994. http://www.ko4bb.com/Manuals/09%29_Misc_Test_Equipment/Thurlby/TTi_TG1010_Operators_Manual.pdf

[117] Traco Power. DC/DC Converters TSR-1 Series, 1 A, 2009. http://docs-europe.electrocomponents.com/webdocs/0ce7/0900766b80ce7679.pdf

[118] Nicolas Tremblay, Zhijie Wang, Bao-Luo Ma, Carl Belec, and Philippe Vigneault. A comparison of crop data measured by two commercial sensors for variable-rate nitrogen application. *Precision Agriculture*, 10(2):145–161, 2009. ISSN 1385-2256. http://dx.doi.org/10.1007/s11119-008-9080-2

[119] Nicolas Tremblay, Zhijie Wang, Bao-Luo Ma, Carl Belec, and Philippe Vigneault. A comparison of crop data measured by two commercial sensors for variable-rate nitrogen application. *Precision Agriculture*, 10(2):145–161, 2009. ISSN 1385-2256. http://dx.doi.org/10.1007/s11119-008-9080-2

[120] Ronald Trostle. Global Agricultural Supply and Demand: Factors Contributing to the Recent Increase in Food Commodity Prices. Technical report, United states department of agriculture, 2013. http://www.ers.usda.gov/publications/wrs-international-agriculture-and-trade-outlook/wrs-0801.aspx

[121] Susan Twining and James Clarke. Future of UK winter oilseed rape production, 2009. http://www.voluntaryinitiative.org.uk/_Attachments/resources/1152_S4.pdf

[122] AJ Wagstaff, NP Merricks, Authorised By, and MJ Ashman. Estimating the utilisation of key licence-exempt spectrum bands. *Mass Consultants Ltd, Tech. Rep*, (3), 2009.

[123] Tim Wark, Peter Corke, Pavan Sikka, Lasse Klingbeil, Ying Guo, Chris Crossman, Phil Valencia, Dave Swain, and Greg Bishop-Hurley. Transforming Agriculture through Pervasive Wireless Sensor Networks. *IEEE Pervasive Computing*, 6(2):50–57, 2007. ISSN 1536-1268. http://dx.doi.org/10.1109/MPRV.2007.47

[124] Simon Andrew Watson. *Mobile Platforms for Underwater Sensor Networks*. PhD thesis, University of Manchester, 2012. https://www.escholar.manchester.ac.uk/uk-ac-man-scw:179880

[125] Mark A. Weissberger. An initial critical summary of models for predicting the attenuation of radio waves by trees. Technical report, DoD Electromagnetic Compatibility Analysis Center, Annapolis, Maryland, 1982. http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA118343

[126] West, J. S. Dr. Jon West Senior Post-doctoral Scientist. http://www.rothamsted.ac.uk/ppi/staff/jsw.html

[127] World Bank. Food Price Watch, July 2013. http://www.worldbank.org/content/dam/Worldbank/document/Poverty%20documents/FoodPriceWatch_July2013.pdf

[128] XP Power. *80-160 Watts JPM Series*, 2009. http://www.xppower.com/pdfs/SF_JPM.pdf

[129] Chenghai Yang, J.H. Everitt, Qian Du, Bin Luo, and J. Chanussot. Using High-Resolution Airborne and Satellite Imagery to Assess Crop Growth and Yield Variability for Precision Agriculture. *Proceedings of the IEEE*, 101(3):582–592, 2013. ISSN 0018-9219.

[130] H. Zaghloul, G. Morrison, and M. Fattouche. Frequency response and path loss measurements of indoor channel. *Electronics Letters*, 27(12):1021 –1022, june 1991. ISSN 0013-5194. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=76210

[131] Chunhua Zhang and John M. Kovacs. The application of small unmanned aerial systems for precision agriculture: a review. *Precision Agriculture*, 13(6):693–712, 2012. ISSN 1385-2256. http://dx.doi.org/10.1007/s11119-012-9274-5

[132] Zhuohui Zhang. Investigation of Wireless Sensor Networks for Precision Agriculture. In *2004 ASAE/CSAE Annual International Meeting*, 2004. http://elibrary.asabe.org/azdez.asp?JID=5&AID=16175&CID=can2004&T=2

[133] Huma Zia, Nick R. Harris, Geoff V. Merrett, Mark Rivers, and Neil Coles. The impact of agricultural activities on water quality: A case for collaborative catchment-scale management using integrated wireless sensor networks. *Computers and Electronics in Agriculture*, 96(0):126 – 138, 2013. ISSN 0168-1699. http://www.sciencedirect.com/science/article/pii/S0168169913001063

[134] Z. Zivkovic, D. Senic, C. Bodendorf, J. Skrzypczynski, and A. Sarolic. Radiation pattern and impedance of a quarter wavelength monopole antenna above a finite ground plane. In *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, pages 1–5, 2012. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6347650

Appendix A

Side work for wines 2 WSN4IP

The WSN4IP sensor node was designed for the WINES 2 sensor project [27]. This project involves the use of wireless sensors in grain silos. It was also considered to use the same backplane system in the WINES 3 underwater sensor node project but this did not happen in the end [124]. The system consists of a plastic sphere of diameter 14 cm which contains a backplane system. The backplane itself is a simple passive device that simply connects all pins of the connectors to the corresponding pins of the other connectors. In the backplane are placed boards with the main processor, sensor interfacing, and the battery management system. Further boards were planned for UWB positioning but these have not been implemented yet. A block diagram is shown in Figure A.1 with currently implemented boards and their connections shown by solid lines while parts that were planned but have not yet been developed are shown with dotted lines.

This appendix only details the parts of the node the author has worked on. For further information on the system as a whole see the research paper in Appendix B or for further

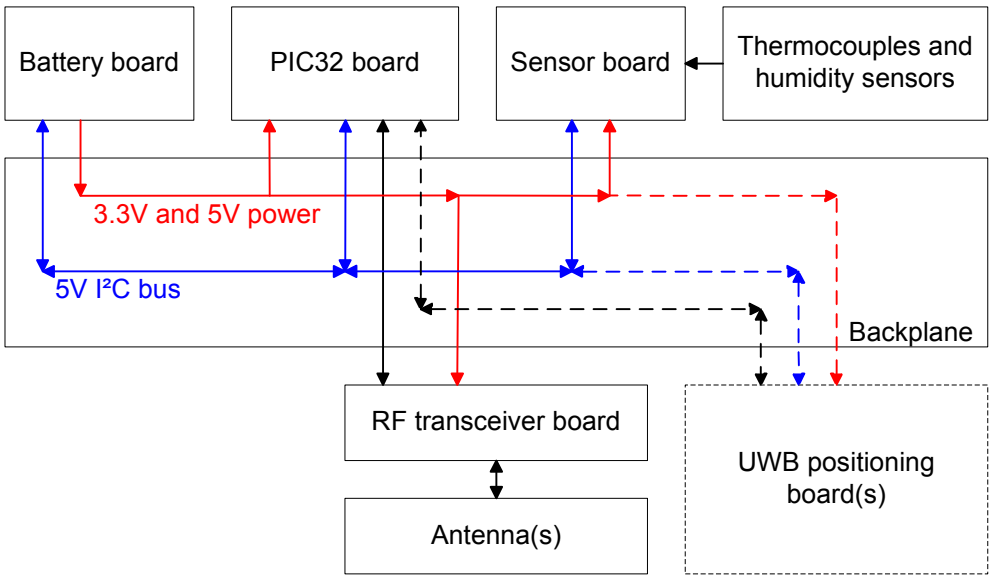


Figure A.1: Block diagram of WSN4IP sensor node

details see Dominic Crutchly's thesis [27].

A.1 Battery board

The battery board was developed to provide power for the Wines 2 WSN4IP sensor node This board is intended for nodes with relatively high power drain and as such is based on a 2Ah lithium polymer battery pack and switched mode converters.

The board contains the battery pack, a charger IC (TI BQ24014), a power switch-over and discharge limiting system, two switching regulators (first a boost to 5V then a buck to 3.3V) and an ADC for monitoring battery voltage. The charger design was based on the Varta Easypack evaluation kit. The switchover circuit is a completely custom design. The regulators were designed using National Semiconductor's on-line "webbench" tool. A picture of the board can be seen in Figure A.2 and a block diagram can be seen in Figure A.3.

Revision A of the board had a design flaw in the switch-over circuit rendering it unusable, specifically that no account had been taken of the body diodes in the mosfets used for switching. Revision B was never fully built and used only as a test platform to check certain aspects of said switch-over circuit. Revision C is considered usable but still has a number of design issues. Firstly the quiescent current of the regulators is much higher than desirable. Secondly the discharge limiting system has no hysteresis which causes the output to oscillate rapidly on and off as the battery reaches the threshold before finally settling in the off state as the battery further discharges. Thirdly other than removing the battery there is no way to switch the board off. Fourthly sometimes the board gets into a state with both the battery and bypass transistors on which should not be able to happen. This seems to be caused by out of specification behaviour from the comparator and logic gate leading to a gate voltage that is not high enough to turn off the bypass transistors but is high enough to trigger the next gate and turn on the battery supply transistors. One board was seen doing this all the time and was fixed by replacement of both logic gate and comparator but other boards have been seen doing it sometimes.

Several possible fixes have been identified for this issue but none have yet been implemented. One fix would be to add a diode such that it is completely impossible for the battery to back feed the charger system. Another option would be to look for mosfets with a higher threshold voltage. Yet another possible fix may be to try adding a pullup to the gate of the mosfet that is failing to turn off.

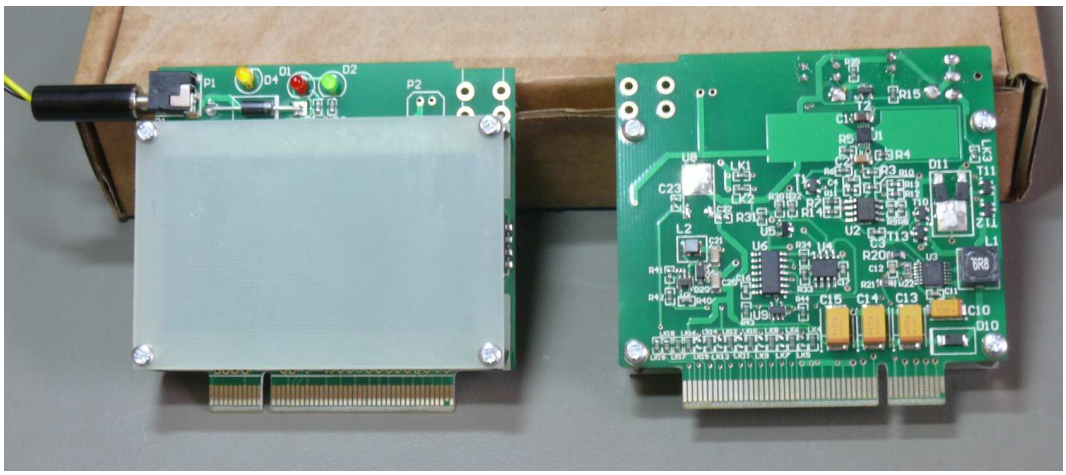


Figure A.2: WSN4IP battery board, top and bottom view

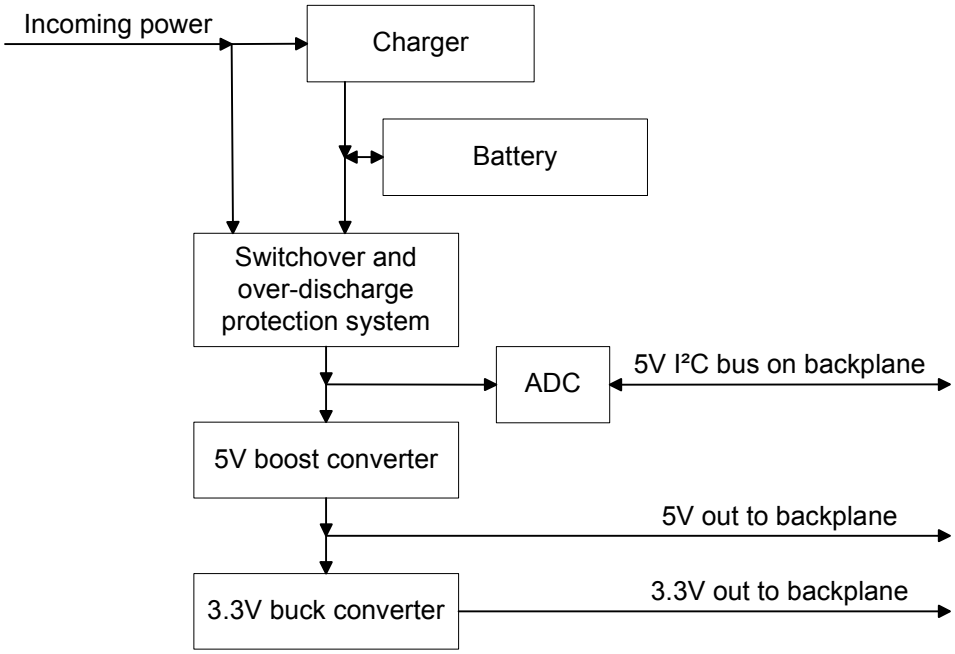


Figure A.3: Block diagram of Battery board

A.2 Bulk charging station

The battery board charges off a 5 V supply and draws a current of just under 1A during the main phase of charging. Individual boards can be charged easily off a lab PSU or off the plug top power supply that came with the Varta Easypack evaluation kit but given that it is likely to take around 3 hours to get a complete charge, charging boards one at a time is not a reasonable option. Therefore a bulk charging station was developed to allow charging of up to 40 boards at a time. DC power at 5V is supplied by a commercial 5V power supply (XP POWER - JPM160PS05) rated to 40A when used under typical office conditions with a 240V supply[1]. The power supply is connected via 4x2.5mm wires (two positive and two negative) to a custom made PCB which splits the output to 40 output connectors. Each individual output is protected by a 1.25A quickblow fuse. The fuses are covered by a perspex cover to prevent accidental shorting of the high current lines from the PSU. The charging station has been used to charge 8 boards at a time at full charge rate with no issues (more boards have been connected at once but only for top-up charging).

The charging station is mounted on a PVC base. A PVC cover attached to the end of the PSU covers the connections to the power supply terminals to ensure no contact is possible with the mains connections. An IEC inlet mounted on the aforementioned cover supplies mains power to the PSU. A picture of the charging station can be seen in Figure A.4.

Figure A.4: Bulk charging station

[1] the headline rating is 32A but according to the data sheet "Max output increases to 125% of rated power when input voltage is between 180 VAC and 264 VAC and ambient temperature is less than +50°C" [128]

A.3 Sensor board

The sensor board is designed to interface to up two thermocouples and up to 8 digital humidity sensors. It has a PIC18f2620 on board to manage reading of the sensors. Associated with the board is a special breakout board that is used to program the PIC on the board and to allow it to be tested outside the backplane system by connecting a PC to the PICs USART (via a level shift IC) and a PICkit Serial Analyzer to the I2C bus that the pic uses for communication with the main processor. When this board is connected the number of supported humidity sensors is reduced to 6. Sufficient software to confirm the boards basic functionality has been written and there are no known design issues with the board. There was however a small design issue with the breakout board which was fixed with a wire-mod.

Due to confusion over sensor types this board is referred to in some places (including writing printed on the board itself) as the "pressure and thermocouple interface". The sensor board and the associated breakout can be seen in Figure A.5.

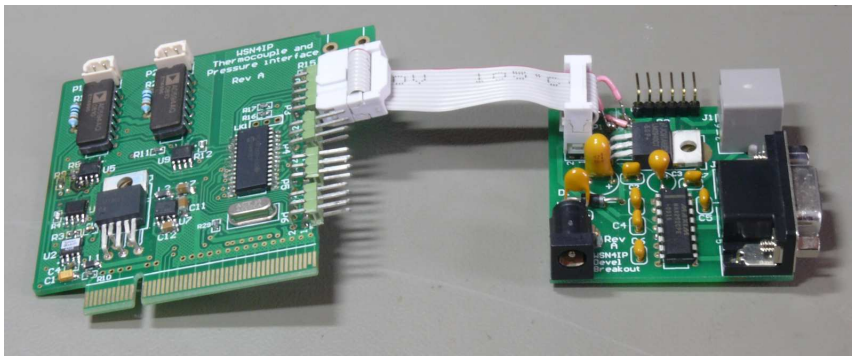


Figure A.5: Sensor board (left) and associated breakout board (right)

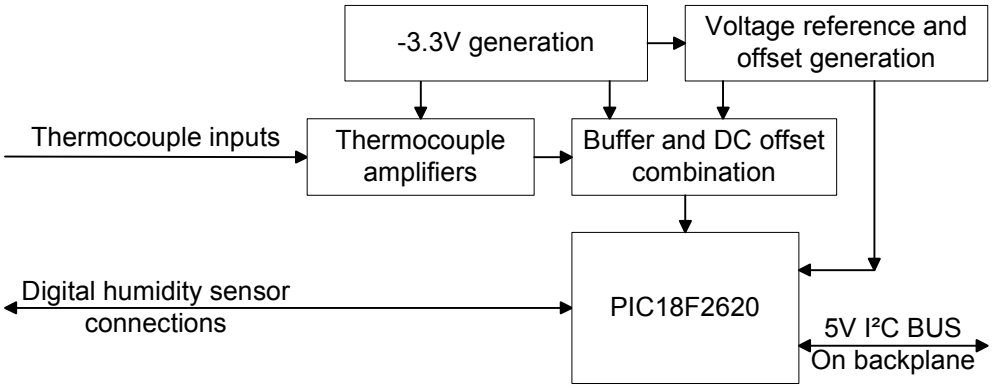


Figure A.6: Block diagram of sensor board

Appendix B

Environmental monitoring in grain paper

This paper provides further information on the WSN4IP backplane system mentioned in chapter A. It was published in the proceedings of Instrumentation and Measurement Technology Conference (I2MTC), 2010 IEEE Page(s): 939 - 943. Its Digital Object Identifier is 10.1109/IMTC.2010.5488046.

Whilst not directly involved in the writing of the paper, this author's name was included to acknowledge his technical contributions to the design, fabrication and testing of the power management and sensing boards in the wireless sensor network node.

Environmental Monitoring in Grain

Graham Parkinson, Dominic Crutchley, Peter M Green, Michalis Antoniou, Mathew Boon, Peter N. Green, Peter R. Green, Robin Sloan and Trevor York
School of Electrical and Electronic Engineering
University of Manchester
Manchester, UK
E-mail: g.parkinson@manchester.ac.uk

***Abstract*—It is shown that Wireless Sensor Networks (WSNs) are capable of deployment in industrial processes which present particularly hostile RF environments. The techniques which have been developed have generic applicability, but in this work they are focused on the monitoring of grain storage. Determination of the environmental conditions in a grain silo is challenged by the dielectric properties of the grain and the multi-path nature of RF propagation in the silo. These challenges apply to both the inter-node WSN communication and node positioning. The methods adopted to meet these challenges and significant results achieved are presented.**

Keywords-wireless sensor networks; local positioning; industrial processes; grain storage; process monitoring; UWB

I. INTRODUCTION

Industrial processing plants invariably present hostile environmental conditions for in situ real-time monitoring of the process conditions. Wireless sensor networks (WSNs), in which the sensor nodes move dynamically and are in intimate contact with the process, provide the ultimate goal for such monitoring but the restrictions presented by the environment must be overcome. Networks of smart mobile sensors distributed throughout the process vessels would provide the ability to communicate and co-operate and so provide a comprehensive picture of the process information. This monitoring then provides the means of process control which will improve process efficiency by reducing raw material, energy usage, wastage (thus minimizing the environmental impact) and also remove the need for close and dangerous occupational exposure to the process.

In order to examine the material distribution and dynamics of a process within a vessel, an imaging method to interrogate the process must be devised. Electromagnetic wave interrogation is possible using, for example, X-rays and magnetic resonance imaging but these methods suffer from a slow imaging rate and they are expensive, potentially dangerous and immobile. Other interrogation methods have emerged, such as acoustic, photonic, gamma rays and electrical excitation including tomographic techniques. These methods are non-invasive but for that very reason they do not provide intimate contact with the process materials and therefore cannot yield deep knowledge of the process conditions.

The environment in which the wireless sensor network operates can present a major technical challenge. Many previous efforts involve communication in air using RF. Implementations also exist for underwater applications [1], and environmental monitoring [2], [3]. Industrial monitoring typically concerns sensors attached to the outside of the plant, for example the monitoring of electrical systems [4], monitoring water pumping stations [5] and industrial control [6].

The WSN techniques developed in this work are aimed at providing small sensors in intimate contact with the process material and will thus provide deeper knowledge of the process status and dynamics, whilst remaining as non-invasive as possible.

The WSN described in this paper is applicable to a wide range of industrial situations. For example, the measurement of temperature, pressure, humidity, impedance and pH are of particular interest in the chemical process industry but would also have applicability in fluid conveyance situations such as oil pipelines in the petrochemical industry.

As part of this WSN research, a technology demonstrator for the monitoring of grain in storage silos has been developed. This first demonstrator will readily exercise the monitoring and wireless networking capability of the system, both in terms of communication and positioning but also presents its own unique challenges. Grain typically has a significant moisture content and this restricts the communication spectrum that can be used for inter-sensor data transmission. This results in an upper frequency in the low microwave region for realistic transmission lengths [7]. Accurate location of the sensors using RF techniques is also necessary and the moisture content of the grain presents similar challenges to this important aspect of the system. Also, grain silos will have discontinuity boundaries of dielectric constant where their walls meet the grain and this will present multiple reflections of the signals which are used for positioning purposes.

This paper describes the development of the WSN and positioning system with generic application but specifically tailored to be deployed in grain. The wireless sensor nodes, based on a PIC32 microcontroller, have been designed for real time monitoring of grain temperature, humidity and sensor orientation. Hardware and software aspects are presented and early results given. Results for sensors in grain incorporating positioning circuit transmitters are also described and results presented showing position accuracy to within the volume of the node.

This work was sponsored by the Engineering and Physical Science Research Council (EPSRC), UK, Grant no.EP/D076900/1)

II. WIRELESS SENSOR NETWORK

A. System Overview

The wireless sensor node consists of a plastic sphere of diameter 14 cm containing the system shown in Fig.1. The system comprises a PIC32 microcontroller [8] with 2 GBs flash memory for measurement data storage. A custom RF circuit provides UWB positioning capability. A Chipcon CC1101 868 MHz transceiver [9] provides data communication between nodes. Temperature, humidity, inclinometer, and 3-axis accelerometer/angular rate sensors are interfaced with the microcontroller. The node is powered by a lithium iron battery with inductive charging capability.

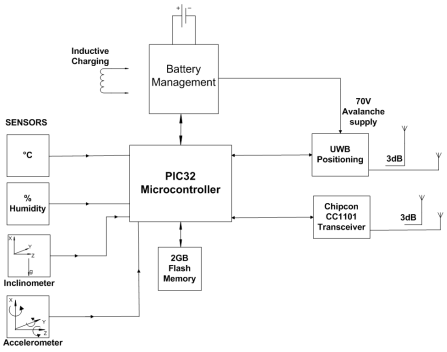


Figure 1. Wireless network sensor – system major components

B. Data Communications

Wheat kernels have a significant unbound water content and a complex permittivity of approximately (3+j0.3) [10]. Measurements have been undertaken [11] to determine the losses in wheat and have confirmed that the path loss determined by a link budget calculation, shown in Fig. 2, is suitable for loss estimation at the frequencies shown. With approximately 10 dB loss per meter at 1 GHz the chosen communication transmission band has been chosen to be 868 MHz which represents an acceptable compromise between achievable data rate and received transmission power.

Data communications are achieved using a Chipcon CC1101 transceiver which interfaces with the microcontroller using a high speed serial bus. The node is equipped with four helical monopole antennas which are connected to the CC1101 though a microwave coupler. The antennas are arranged to achieve node to node communication irrespective of the relative orientation of the nodes.

C. Network Architecture and Communication Protocols

The basic function of the network is to relay data from all nodes within the silo back to a central base station. Also the network must be able to operate within the lossy medium and during periods where grain movement creates high node mobility.

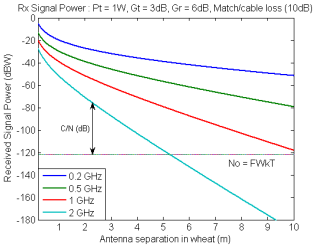


Figure 2. Link budget estimation of carrier to noise ration for WSN antennas in wheat for varyimg transmission frequencies

The power budget for each node prevents the use of high transmission power to overcome the losses in the grain. As a consequence, ad hoc networking has been chosen to reduce the overall transmission power for each packet, whilst allowing all the nodes to remain in contact with the base station.

Without the assistance of a base-station or network infrastructure, each node will function as a router in the network, allowing messages to be passed from node to node until the destination is reached. However, although ad hoc networking allows us to resolve the problem of communicating across the entire silo, it brings other problems. It is expected that the network will sometimes be operating whilst grain is moving within the silo. As a result, the network topology is expected to change, sometimes abruptly, as nodes move with the grain. The system is expected to tolerate both very low rates of movement (all the way down to a static condition), and very high rates of movement (e.g. rapid emptying of silos).

This type of network topology change brings a new challenge, as multiple nodes of the network are expected to rapidly undergo multiple topology changes. This situation places large demands on the routing protocols and poses a serious challenge to the operation of low power MAC [12] or Link Layers protocols, which to some extent assume a low level of disruption to network topology. Current protocols for wireless ad hoc sensor networks are typically designed to cope only with the static condition or situations where one or two nodes move at a slow speed (e.g. walking pace) through an otherwise static network.

Moreover, little work on fully mobile networks is done within the WSN field. Instead, one must look to the MANET community as a source of promising routing protocols designed to deal with constant topology change. The downside of this approach is that such protocols assume a much larger power budget, and are ill suited to operation on low power WSN nodes.

Communications protocols are implemented through software running on the PIC32 and are arranged in a traditional layered protocol stack [13]. To control the performance of the ad hoc and mobility requirements of a node, a cross-layer architecture [14] is added to allow for inter-layer communications between the different protocols.

In addition, the cross-layer architecture is extended beyond the communications protocols to become a 'cross-application' architecture, capable of facilitating communication between any application or application component running on the node.

III. Positioning

The difficulty in electromagnetic propagation in wheat due to its moisture content has already been discussed with respect to node to node communication. Similar considerations apply to the signal propagation for positioning purposes and attenuation of >10dB/m above 1 GHz renders conventional techniques, such as GPS, inoperable.

A technique which has been proven successful by the authors is that of time difference of arrival (TDOA) ultra-wideband (UWB) positioning [15], [16]. This has now been adapted to provide a robust positioning system and positioning circuitry has been integrated into the WSN node housing.

A. Methodology

Figure 3 shows a schematic of the wheat silo demonstrator with nodes containing avalanche pulse generators embedded in the grain. The pulse generator (see inset Fig. 3) consists of a 1 MHz crystal oscillator and a ÷2 frequency divider driving an avalanche pulse generator at 500 kHz. This pulse has a width of about 0.5 ns and is thus capable of delivering the UWB signal.

In order to provide a sufficiently omni-directional transmit signal from the node, which also contains the wireless system electronics, two orthogonally positioned (in terms of their polarization) ring monopole antennas are positioned in the node, as shown in Fig. 4.

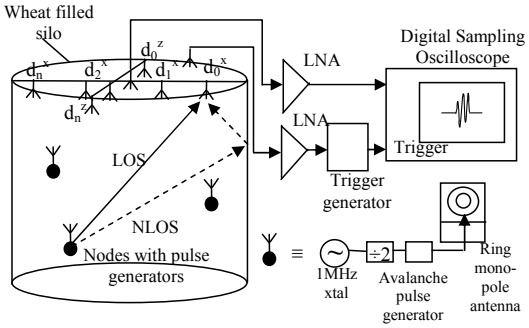


Figure 3. 3D UWB positioning using avalanche pulse generators buried in a wheat silo.

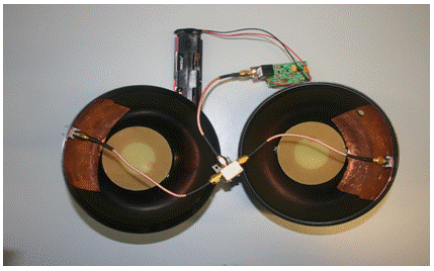


Figure 4. Ring monopole antennas and pulse generator in spherical node

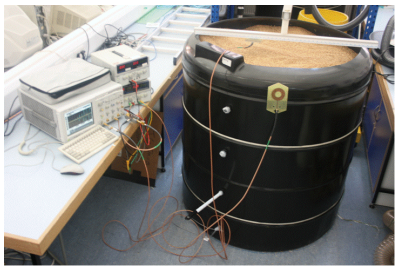


Figure 5. Experimental rig

The experimental demonstrator is shown in Fig. 5 where the silo model is 1.16 m diameter by 1.29 m deep. Pulse transmissions from the nodes are received by an array of antennas on top of the silo and the received signals are amplified by a low noise amplifier (LNA) and displayed and stored on a 20 GSa/s digital sampling oscilloscope (DSO). A further antenna, LNA and fast pulse generator provide a trigger pulse for the DSO to retain temporal coherence.

A range gate is applied to the received pulse to isolate the line of sight signal from the multipath components and the TDOA of the signals arriving at each antenna in the antenna array is then calculated using a MATLAB program [15]. The intersection points of spherical functions relating to each TDOA are then calculated.

An asynchronous positioning method is desired and algorithms capable of achieving this have been developed [15], [16]. This method relies upon an initial estimate of the first spherical function radius which generates a spread of intersection points of all the spherical functions. These intersection points will converge to a single location as the true value of the first spherical function radius is approached. A standard deviation cost function defines the RMS distance between the scattered intersection points.

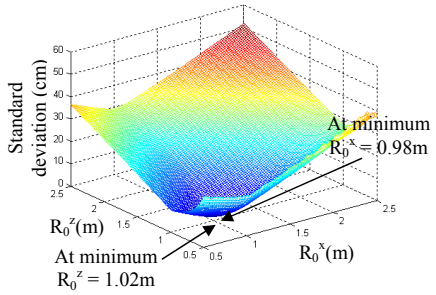


Figure 6. RMS variation of intersection point scatter about mean showing true position

Figure 7. Flexible ring monople antenna with conformal flexible ground plane

B. Results

The standard deviation cost function plane is plotted against R_0^x and R_0^z and yields a sharp minimum at the true position estimate. A typical result is shown in Fig. 6 where the radii of the first spherical functions (R_0) are determined. Further mathematical manipulation in MATLAB yields the x, y, z co-ordinates of the estimated position and for these tests is found to be within the volume of the sphere for the example shown.

IV. System Integration

Ring monopole antennas [16], [17] have been shown to be suitable for integration into the node sphere housing. Fig. 4 shows two such antennas fabricated from FR4. Improvements have been made to overcome the rigidity of the FR4 and aid the manufacturing and assembly process by implementing the antennas in flexible PCB. This, along with a flexible ground plane, allows the whole antenna assembly to be conformally mounted to the inside of the node sphere. An example prior to assembly is shown in Fig. 7.

Figure 8. Backplane in wireless sensor node

The system shown in Fig. 1 has been implemented in modular form using a backplane and a set of PCB cards. These cards comprise PIC32 microcontroller, CC1101 transceiver data communications, UWB positioning, sensor interface and battery power management. Fig. 8 shows the node, open, with the backplane included and slots for card placement.

V. Conclusions

A wireless sensor network has been developed for generic use in industrial processes. It has been applied to the specific case of wheat in a silo and the challenges of wheat moisture content and multipath propagation have been mitigated. Working demonstrators of the wireless network and the positioning system have been established and these show that future integration and deployment within an industrial silo is feasible.

Acknowledgment

The authors would like to thank Dr. J. D. Davis, BAE Systems, for his initial work on this project.

References

[1] I.F. Akyildiz, D. Pompili, T. Melodia, "Underwater acoustic sensor networks: research challenges," *Ad Hoc Networks* 3 (2005) pp. 257 – 279.

[2] M. Kuprilehto, M. Hannikainen and T.D. Hamalainen, "A survey of applications in wireless sensor networks," *EURASIP Journal on Wireless Communcations and Networking*, 5, 2005, pp. 774 – 788.

[3] K. Martinez, R. Ong and J. Hart, "Glacsweb: a sensor network for hostile environments," *2004 First Annual IEEE Communications Society Conf. On Sensor and Ad Hoc Communications and Networks*, Oct. 2004, pp 81 – 87.

[4] F. Salvadori, M. de Campos, R. de Figueiredo, C. Gehrke, C. Rech and P.S. Sausen, " Monitoring and diagonsis in industrial systems using wireless sensor networks," *IEEE Int. Symp. on Intelligent Signal Processing*, 2007, pp 40 – 45.

[5] G.V. Merrett, N.R. Harris, B.M. Al-Hashimi and N.M. White, "Energy controlled reporting for industrial monitoring wireless sensor networks," *IEEE Sensors,* 2006, pp. 892 -895.

[6] H. Ramamurthy, B.S. Prabhu, R. Gadh and A.M. Madni, "Wireless industrial monitoring and control using a smart sensor platform," *IEEE Sensors Journal*, vol. 7, no. 5, May 2007, pp 611 – 618.

[7] P. Armstrong, "Wireless data transmission of network sensors in grain storage," ASAE Annual Meeting, ASAE Riviera Hotel and Conference Center, Las Vegas, USA, 27– 30th July 2003.

[8] "PIC32MX3XX/4XX Family Data Sheet", Data Sheet, MicroChip, 2009.

[9] "CC1101 Low-Cost Low-Power Sub-1GHz RF Transceiver (Enhanced CC1100),", Manual, Texas Instruments.

[10] S. O. Nelson, "Review and assessment of radio-frequency and microwave energy for stored-grain insect control," *Trans. of the ASAE*, vol. 39(4), pp. 1475- 1484, 1996.

[11] M. Antoniou, M.C. Boon, P.N. Green and T.A. York, "Wireless sensor networks for industrial procesesses," *IEEE Sensors Applications Symp.* New Orleans, USA, Feb. 2009.

[12] W. Ye, J. H. and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," *Proc. INFOCOM 2002 21st Annual Joint Conf. of the IEEE Computer and Communications Socs*.,vol.3, pp. 1567-1576, 2002.

[13] "Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model," BSI standard BS EN ISO/IEC 7498-1:1995

[14] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *IEEE Communications magazine*," vol.43, no. 12, pp.112-119, Dec. 2005.

[15] J.G. Davis, R. Sloan, A. J. Peyton and M. Bilal "A positioning algorithm for wireless sensors in rich multipath environments," *IEEE Wireless and Components Letters,* vol. 18, no. 9, pp 644-646, Sept. 2008.

[16] G. Parkinson, M.C. Boon, J.G. Davis and R. Sloan, "3D positioning using spherical location algorithms for networked wireless sensors deployed in grain," *IEEE MTT-S Int. Microwave Symp*. Dig., Boston, MA, June 2009, pp. 1417-1420

[17] J. Liang, C.C. Ciau, X. Chen and C.G. Parini, "Printed circular ring monopole antennas," *Microwave and Optical Technology Letters*, vol.45, no.5, pp 372-375, June 2005.

Appendix C

Using RF for sensing

As well as using RF for communication between nodes it was suggested by P.R. Green that it could actually be used for sensing. In particular if a pair of closely spaced antennas is passed over a line of crops such that the plants pass between the two antennas it may reveal things about that line of crops (whether there is a plant in the expected location or not, maybe other information too).

Such a system is clearly only feasible in crops that grow in neat rows with spaces between them. It would not be feasible in a crop like rapeseed which grows into a continuous mass of intertwined plants.

To provide good spacial resolution and allow for multiple sensors to operate close together while keeping the sense head size manageable a higher frequency than is typically used for wireless communications is needed. The region around 18 GHz (specifically 16 GHz to 20 GHz) was chosen as a compromise between availability of equipment and ability for antennas to be physically small.

The first step in determining the feasibility of such a system was to perform simulations. The HFSS simulation software was used for these. A HFSS model of a possible sense head was designed with two simple antennas (each modelled on a very simple waveguide transition) pointing at each other and then columns of water placed between them to represent the foliage. Simulations were carried out with three scenarios. No columns of water, a single 0.5 mm radius column of water and a group of sixteen[1] 0.5 mm radius columns of water. The HFSS model is shown in figures C.1 and C.2[2].

The group of water columns caused a substantial drop in signal of approximately 6-19 dB. However the single water column only caused a drop of approximately 1-2 dB. This should still be detectable but preventing other variations from swamping the change due to the plant material may prove a problem. Best results were seen at a frequency of 16.7 GHz for the single water column case and 16.8 GHz for the multiple water columns case. The results are shown in figures C.3 and C.4.

[1] due to the use of a symmetry boundary only 8 columns were actually present in the model

[2] The large box in the figures is the box that defines the limits of the model, only half of each head is modelled because a symmetry boundary is used.

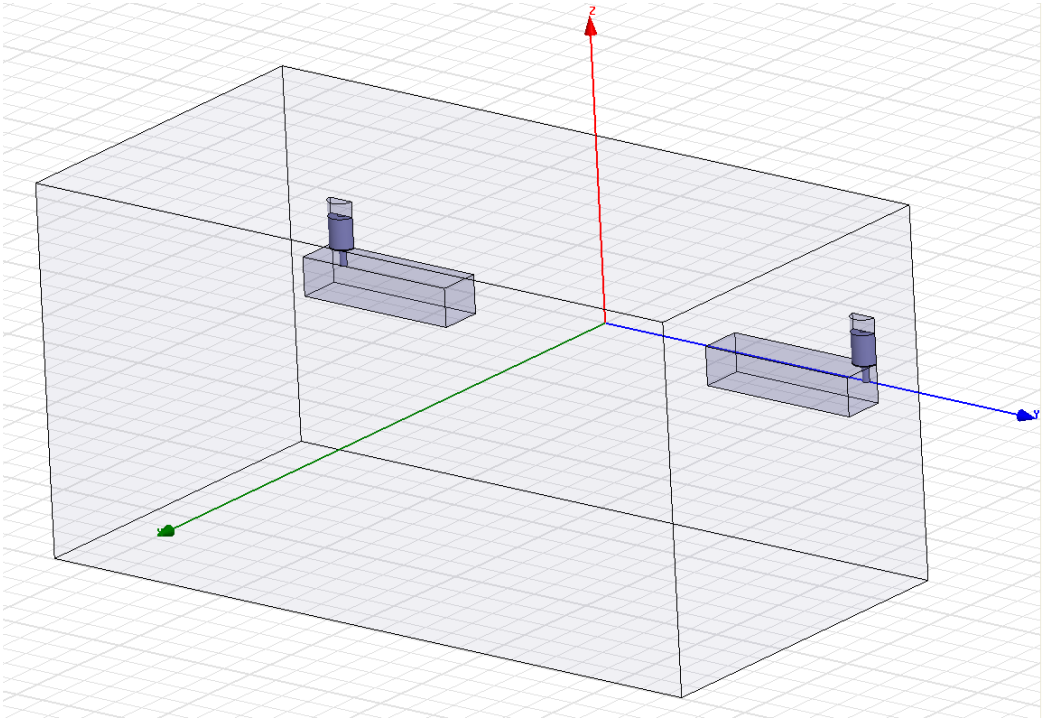


Figure C.1: HFSS model of RF sense heads

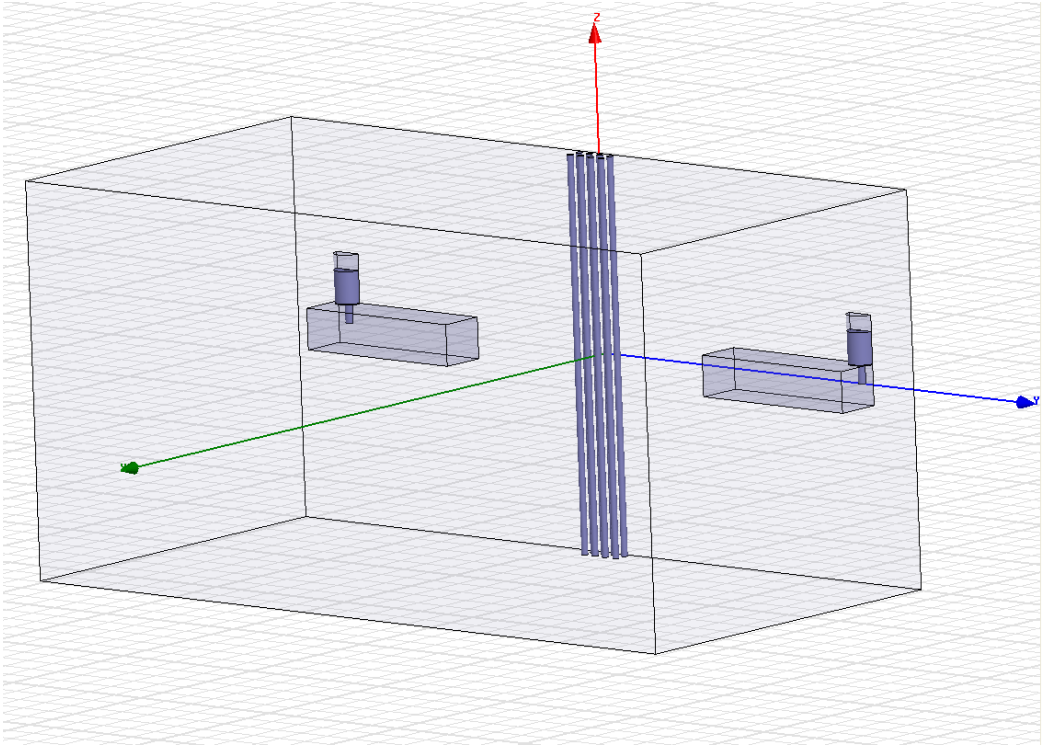


Figure C.2: HFSS model of RF sense heads with water columns between them

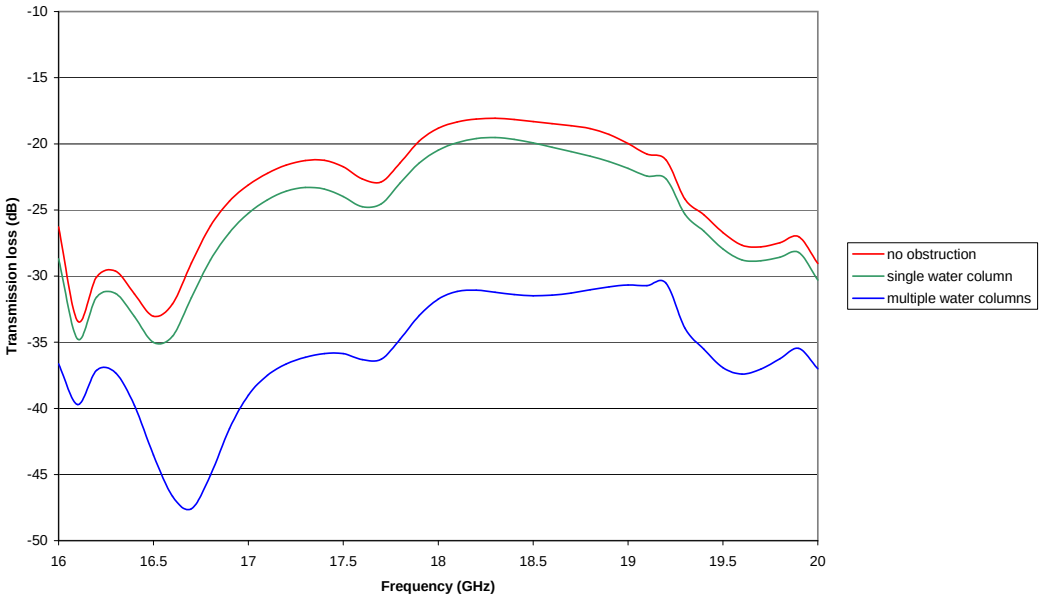


Figure C.3: Results of HFSS simulations to investigate feasibility of RF sensing.

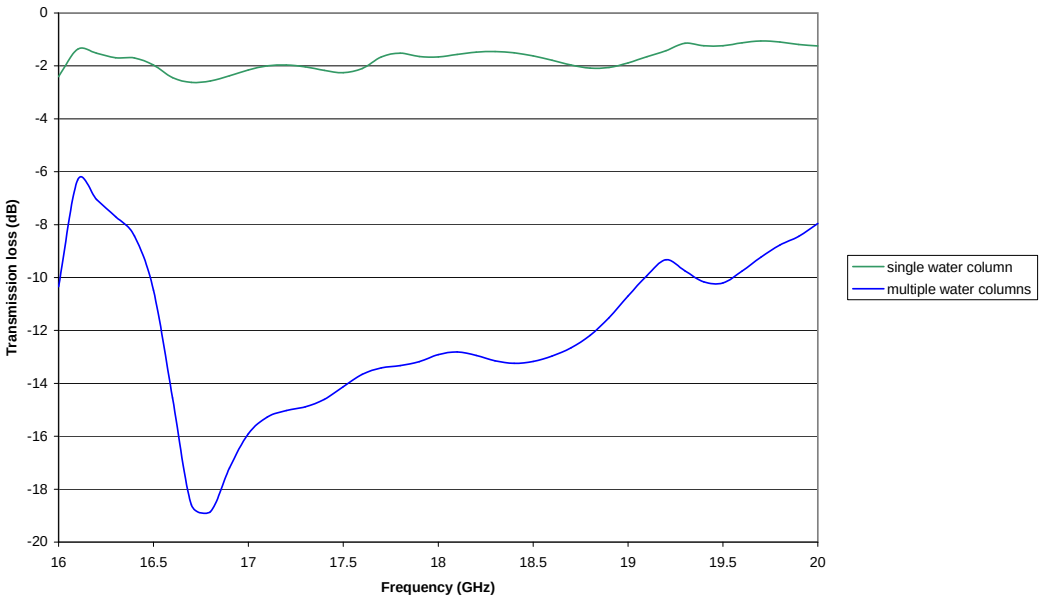


Figure C.4: Results of HFSS simulations to investigate feasibility of RF sensing with results from unobstructed case subtracted.

The next step was to design and have constructed by the mechanical workshop a mount that could hold two waveguide transitions at an appropriate separation while allowing plant material to be passed between them. The mount was built in a U shape out of PVC to allow plant material to be passed between the transitions. The mount is designed to hold the transitions 8 cm apart and there is 35 cm of clear space between the centre of the transitions and the cross bar of the U shape. A picture of the mount can be seen in figure C.5.

Unfortunately the mount is rather flexible. With minor force on the mount the gap between the transitions varies from about 4 cm to 12 cm. The author suspects that the differences in channel loss caused by flexing of the mount may be too great and obscure the differences caused by passing plant material through it. The next step in the development would be to test this on a vector network analyser and if necessary add reinforcement to the mount. Once this is done testing with plant material would then be able to begin.

Unfortunately failures of vector network analysers combined with the fact that this was a side activity and not the main thrust of the project means that these tests have not happened.

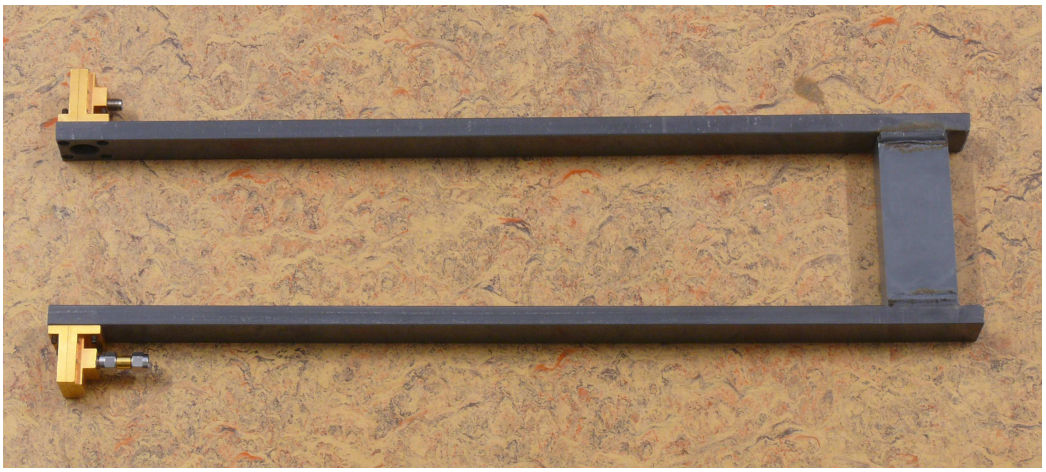


Figure C.5: Waveguide transitions mounted on U shaped frame intended to be used for RF sensing experiments.

Appendix D

Four port resistive splitter

While working on the experimental set-up for link budget measurements described in chapter 7 a four port splitter was designed. This device serves to provide tap off points on the simulated channel that can be used for monitoring the channel, and potentially for injecting noise or hooking up a third transceiver for multi-hop testing. If only three ports are being used then the fourth port is terminated with an off the shelf SMA terminator. The splitter was designed to be symmetrical such that signals introduced on any port would be available on all the ports. Off the shelf splitters were investigated but all the splitters the author found were of asymmetric designs.

To produce the splitter the ports are connected to a central point. If the ports were directly connected to the central point then there would be an impedance mismatch as a single $50\,\Omega$ line would be driving three $50\,\Omega$ lines (an overall impedance of $16\frac{2}{3}\,\Omega$) in parallel. To avoid the impedance mismatch a resistor was inserted in each line to the central point. To keep the design simple and to meet the objective of a symmetrical design it was decided to make all resistors equal value.

To calculate the resistor value, basic circuit analysis techniques were used. For the purposes of this analysis all but one of the external connections were replaced with resistors matching desired characteristic impedance. The overall resistance was then set equal to the desired characteristic impedance. Analysing this circuit resulted in Equation D.1 where Z_0 is the characteristic impedance of the line and n is the number of ports.

$$\frac{Z_0 + R}{n - 1} + R = Z_0 \tag{D.1}$$

By inserting the $50\,\Omega$ characteristic impedance of standard coax components and the 4 ports desired for the splitter and solving the equation is is possible to calculate a value for R as shown in Equations D.2 and D.3.

$$\frac{50 + R}{3} + R = 50 \tag{D.2}$$

$$R = 25\,\Omega \tag{D.3}$$

Ordinary surface mount 1 % 0603 chip resistors with a resistance of 24.9 Ω (the closest available value to 25 Ω) were used to make the splitter. The four way split was kept as small as deemed practical for hand construction to minimise the impact of transmission line effects inside the splitter. The PCB was 1.6mm FR4 with 1 oz copper. With this material 3mm wide lines are needed for 50 ohm impedance [61, pp 166]. Lines of this width were used to connect from the SMA connectors to the resistors surrounding the central split. The resistive splitter is pictured in Figure D.1.

Theoretically a four port splitter of this design, if the resistors were perfect resistors with a resistance of 25 Ω and the construction were infinitely small, would have a loss of 9.54 dB and would cause no reflections. In practice at 2.465213 GHz (the closest data point of 2.45 GHz that was in the VNA trace) the return loss was around 8.6 dB and the propagation loss was around 10.1 dB. A selection of the S-parameters can be seen in Figure D.2. While far from perfect this performance is acceptable for the intended application

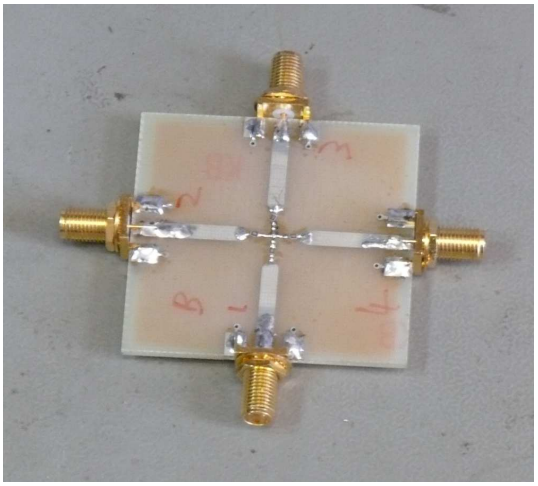


Figure D.1: Photograph of the 4-port resistive splitter

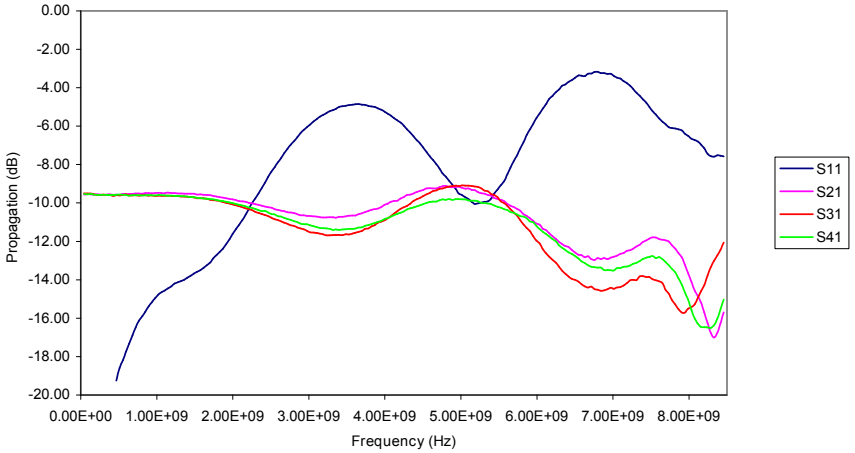


Figure D.2: Selected S-parameters of the 4-port resistive splitter

Appendix E

Contents of the optical disc

An optical disc is provided to the examiners with additional material. It is structured as follows.

The directories “ampmeasurement”, “fieldmeasurement”, “fieldmeasurementprocess”, “gateway”, “miwi”, “powermeasurement” and “powermeasurementprocess” contain software developed for the project. Some of this software was developed from scratch and other software was developed by modifying existing example code. Further information is available in Appendices F to M.

The directories “18-5reg”, “18-15reg”, “basepower-revb”, “batteryboard-revc”, “nodehardware-revb”, “rfmb-revb”, “usbisolator”, “pressurethermointerface_reva” and “wsnpowermeasurement-revb” contain PCB designs developed as part of the project. Further information is available in Appendices N to S.

The directory “hfss“ contains HFSS models constructed during the antenna investigation described in Chapter 4 and the investigation of using RF for sensing described in Appendix C.

The directory “deploymentrecordings” contains the raw data gathered by the base station during the deployments. The files 120229_1239 to 120307_1050 represent the test deployment at the university. The files 120315_0930 to 120816_0310 represent the deployment at Tatton Dale Farm.

The directory “fieldmeasurementrecordings” contains the raw recordings made using the field measurement system at Rothamsted and Tatton Dale Farm. Note that an error was made with dates when naming the files. The recordings taken on the 26th of July 2012 were mislabeled as 20120626 and the recordings taken on the 2nd of august 2012 were mislabelled as 20120702.

The file “thesis.pdf” contains a pdf copy of the thesis.

Appendix F

Power measurement acquisition source code

This appendix is a place holder for the code for acquisition from the power measurement system. The code was created by expanding the "data translation C example". The actual code is not presented in the thesis itself due to copyright concerns but is available to the examiners on the included disc in the directory "powermeasurement".

Four existing source files were modified. These files were cexmpl.c, cexmpl.h, cexmpl.rc and resource.h. These files are provided both in their complete form after modification and as diffs between their unmodified and modified forms so that the changes that were made can be seen. Two new source files were also introduced: calibration.c and filewrite.c. All files whether modified or newly introduced are likely to contain some code from the data translation example.

Appendix G

Power measurement processing source code

This appendix contains the source code for the processing code for the power measurement system. There were two generations of the code, one written in Matlab and one written in Java. Both were written from scratch by the author.

G.1 Matlab version

In the Matlab version of the code there are two top level files intended to be run one after the other. readdata.m reads the data and processdata.m processes it. A third file filterandtrim.m is used as a function by processdata.m. The code is presented here and is also available on the disc included for the examiners in the directory "powermeasurement".

G.1.1 readdata.m

```
f=fopen(filename);

%read and check file identification
identification=char(fread(f,[1,12],'*uchar',0,'l'));
if strcmp(identification,'WSNPWRMEASUR') == 0
    display('failed to find file identification, did this file really come from the power measurement rig?!');
    fclose(f);
    return;
end

version=fread(f,1,'*uint16',0,'l');
display(['version 0x' dec2hex(version,4)]);

if (version/256) > 0
    display('version too high for this code to read');
    fclose(f);
    return;
end

headersize=fread(f,1,'uint16',0,'l');

header=fread(f,headersize,'*uchar',0,'l');
%read fixed part of header
%note: file format documentation uses offsets, matlab array indexes start from 1 so will be 1 greater than offsets
channels=header(1);
datawordsize=header(2);
descoffset=cast(header(3),'uint16')+(256*cast(header(4),'uint16'));
totalsamplesperchannel=typecast(header(5:8),'uint32');
```

```
zerocalibrationvoltage=typecast(header(9:16),'double');

%read per channel part of header
zerocalibrationreadings = zeros(channels,1);
loadedcalibrationreadings = zeros(channels,1);
calibrationresistorvalues = zeros(channels,1);
loadedcalibrationvoltages = zeros(channels,1);

for i = 1:channels
    cds = ((i-1)*16)+17;
    zerocalibrationreadings(i) = typecast(header(cds:cds+1),'uint16');
    loadedcalibrationreadings(i) = typecast(header(cds+2:cds+3),'uint16');
    calibrationresistorvalues(i) = typecast(header(cds+4:cds+7),'uint32');
    loadedcalibrationvoltages(i) = typecast(header(cds+8:cds+15),'double');
end

%read description from header
desc = header(descoffset+1:length(header));
terminatorpos = find(desc==0,1);
desc = desc(1:terminatorpos);
desc = char(desc.');

data=fread(f,[3,inf],'uint16',0,'l');
fclose(f);
if length(data) ~= totalsamplesperchannel
    display('number of data samples read from file does not match number specified in header');
end

%data=data-32768; %remove offset
%data=data/32768*10; %convert to volts
data=data.'; %i'd rather have columns than rows

%filter=load('10khz 511 tap hanning filter.txt');
filter=load('5khz 511 tap hanning filter.txt');
%filter=load('525hz 511 tap hanning filter.txt');
%filter=load('525hz 4095 tap hanning filter.txt');

%we don't care about the time column of the filter
filter=filter(:,2);
```

G.1.2 processdata.m

```
%get a smaller peice of the data to work with to speed things up
%datatrimmed=data(1:100000,:);
%seperate off the channels
hdata=data(:,3);
mdata=data(:,2);
ldata=data(:,1);
filteredhdata=filterandtrim(hdata,filter);
filteredmdata=filterandtrim(mdata,filter);
filteredldata=filterandtrim(ldata,filter);
%now we have filtered the data we can downsample to a nicer samplerate
filteredhdata=downsample(filteredhdata,10);
filteredmdata=downsample(filteredmdata,10);
filteredldata=downsample(filteredldata,10);

%calculate microaps per ADC unit for each channel
microampsperadcunit = zeros(channels,1);

for i = 1:channels
    microamps = loadedcalibrationvoltages(i)/calibrationresistorvalues(i)*1000000;
        microampsperadcunit(i) = microamps / (loadedcalibrationreadings(i)-zerocalibrationreadings(i));
end

%fix zeros


%scale figures and remove DC offsets so all channels are in microamps
scaledhdata=(filteredhdata-zerocalibrationreadings(3))*microampsperadcunit(3);
scaledmdata=(filteredmdata-zerocalibrationreadings(2))*microampsperadcunit(2);
scaledldata=(filteredldata-zerocalibrationreadings(1))*microampsperadcunit(1);

filteredhrms=sqrt(mean(filteredhdata.*filteredhdata));
scaledhrms=sqrt(mean(scaledhdata.*scaledhdata));
scaledmrms=sqrt(mean(scaledmdata.*scaledmdata));
scaledlrms=sqrt(mean(scaledldata.*scaledldata));

%first try recombination algorithm
%don't trust any value where the unscaled value that is more than 80% of ADC full scale

cdata = zeros(length(scaledhdata),1);
for i = 1:length(scaledhdata);
    if scaledhdata(i,1) < 8
        cdata(i,1) = scaledhdata(i,1);
    elseif scaledmdata(i,1) < 800
        cdata(i,1) = scaledmdata(i,1);
```

```
    else
        cdata(i,1) = scaledldata(i,1);
    end
end

crms=sqrt(mean(cdata.*cdata));

%y=filtereddata;
%frequency spectrum
%Fs = 10000;                    % Sampling frequency
%T = 1/Fs;                      % Sample time
%L = length(y);                     % Length of signal
%t = (0:L-1)*T;                % Time vector
%NFFT = 2^nextpow2(L); % Next power of 2 from length of y
%Y = fft(y,NFFT)/L;
%AMP = 2*abs(Y(1:NFFT/2+1));
%POW=AMP.*AMP;
%f = Fs/2*linspace(0,1,NFFT/2+1);
%plot(f,POW);
%title('Single-Sided Amplitude Spectrum of y(t)')
%xlabel('Frequency (Hz)')
%ylabel('|Y(f)|')
```

G.1.3 filterandtrim.m

```
function result = filterandtrim(data,filter)
%filter the data
result=conv(data,filter);
%compensate for the dc gain of the filter
result=result/4096;
%trim back to original length
trimlen=(length(filter)-1)/2;
startpos=trimlen+1;
endpos=length(result)-trimlen;
result=result(startpos:endpos,:);
```

G.2 Java version

The Java version of the code is contained in a single Java file called “powermeasurementprocess.java”. JFreechart is used for plotting. The code is presented here and is also available on the disc included for the examiners in the directory “powermeasurement”.

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.lang.reflect.Field;
import java.lang.reflect.InvocationTargetException;
import java.nio.ByteBuffer;
import java.nio.ByteOrder;
import java.nio.CharBuffer;
import java.nio.charset.Charset;
import java.util.zip.DataFormatException;

import javax.swing.*;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;

import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.axis.LogarithmicAxis;
import org.jfree.chart.axis.NumberAxis;
import org.jfree.chart.axis.ValueAxis;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.chart.plot.XYPlot;
import org.jfree.data.xy.DefaultXYDataset;

// this call will probablly go, it's only here for now because I haven't figured out how
// i'm going to get rid of it yet
//class ProcessedData {
//
//        public boolean real;
//        public Object title;
//        public String xlabel;
//        public String ylabel;
//        public String[] xdata;
```

```
//        public String[] ydata;
//
//}

class Data {
        public double [][] filtereddata;
        public double[][] scaleddata;
        public double[] combineddata;
        public int decimatedblocksize;
        public int decimatedsamplesperchannel;
        public double zerocalibrationvoltage;
        public String description;
        public int channelcount;
        public byte datawordsize;
        public int[] zerocalibrationreadings;
        public char[] loadedcalibrationreadings;
        public int[] calibrationresistorvalues;
        public double[] loadedcalibrationvoltages;

}

class Graphsettings {

        String fieldname; //name of field in data class
        int channel; //measurement channel (-1 if not applicable)
        String displayname; //display name of field
        boolean logm; //true if log manitude graph
        String yaxistitle; //title for y axis
        public String toString() {
                String s = displayname;
                if (channel >= 0) s +=" channel "+channel;
                if (logm) s += " (logm)"; else s += " (lin)";
                return s;
        }
        public Graphsettings( String fieldname, int channel,
                        String displayname,boolean logm, String yaxistitle) {
                super();

                this.fieldname = fieldname;
                this.channel = channel;
                this.displayname = displayname;
                this.logm = logm;
                this.yaxistitle = yaxistitle;
        }
}

@SuppressWarnings("serial")
public class powermeasurementprocess extends JFrame implements ActionListener, ListSelectionListener {


        private DefaultXYDataset dataset;
        private JFileChooser fc = new JFileChooser("d:\\");
        private File datafile;



        private Data data = new Data();
        private DefaultListModel graphselectorentries;
        private JList graphselector;
        private XYPlot plot;
        private JLabel descriptionlabel;
        private JFreeChart chart;


        public powermeasurementprocess(String[] args) {


                //displayfile("Y:||fieldmeasurementrecordings||fieldmeasurement_20110124corridoortest.dat");

                setSize(500, 500);
                dataset = new DefaultXYDataset();


                chart = ChartFactory.createXYLineChart("no data loaded","time (s)","current magnitude (uA)",dataset,PlotOrientation.VERTICAL,false,false,false);
                plot = (XYPlot) chart.getPlot();



                //setsummarychartrange(data.size());
                //plot.getRangeAxis().setRange(datalowrounded,datahighrounded);
                ChartPanel chartp = new ChartPanel(chart);
                add(chartp);

                descriptionlabel = new JLabel("no data loaded",SwingConstants.CENTER);
                add(descriptionlabel,BorderLayout.NORTH);


                graphselectorentries = new DefaultListModel();
                graphselectorentries.addElement("no data loaded");
```

```
        graphselector = new JList(graphselectorentries);
        graphselector.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

        graphselector.addListSelectionListener(this);

        add(graphselector,BorderLayout.WEST);

        JPanel buttonbar = new JPanel();
        add(buttonbar,BorderLayout.SOUTH);



        JButton loaddatabutton = new JButton("load data");
        loaddatabutton.setActionCommand("loaddatabutton");
        loaddatabutton.addActionListener(this);
        buttonbar.add(loaddatabutton);

        JButton exportbutton = new JButton("export");
        exportbutton.setActionCommand("exportbutton");
        exportbutton.addActionListener(this);
        buttonbar.add(exportbutton);
        setDefaultCloseOperation(DISPOSE_ON_CLOSE);

        JButton exportsummarybutton = new JButton("export summary");
        exportsummarybutton.setActionCommand("exportsummarybutton");
        exportsummarybutton.addActionListener(this);
        buttonbar.add(exportsummarybutton);


        setVisible(true);
}

private void addlistentriesforchannels(DefaultListModel graphselectorentries,
                String fieldname, String displayname,boolean logallowed, String ytitlestart, String ytitleunits) {

        for (int i=0;i<data.channelcount;i++) {
                graphselectorentries.addElement(new Graphsettings( fieldname, i, displayname,false,ytitlestart+" "+ytitleunits));
                if (logallowed) graphselectorentries.addElement(new Graphsettings( fieldname, i, displayname,true,ytitlestart+" magnitude "+ytitleunits));
        }
}

//change the y axis class (to e.g. NumberAxis or LogarithmicAxis) and the y axis label
//while preserving the label font (may be extended to preserve other stuff later
private static void setYAxisType(XYPlot plot, Class<? extends ValueAxis> axisclass, String label) {
        try {
                ValueAxis oldrangeaxis = plot.getRangeAxis();
                Font labelfont = oldrangeaxis.getLabelFont();
                Font ticklabelfont = oldrangeaxis.getTickLabelFont();
                ValueAxis newrangeaxis;
                newrangeaxis = axisclass.getConstructor(String.class).newInstance(label);

                newrangeaxis.setLabelFont(labelfont);
                newrangeaxis.setTickLabelFont(ticklabelfont);
                plot.setRangeAxis(newrangeaxis);
        } catch (IllegalArgumentException e) {
                throw new RuntimeException(e);
        } catch (SecurityException e) {
                throw new RuntimeException(e);
        } catch (InstantiationException e) {
                throw new RuntimeException(e);
        } catch (IllegalAccessException e) {
                throw new RuntimeException(e);
        } catch (InvocationTargetException e) {
                throw new RuntimeException(e);
        } catch (NoSuchMethodException e) {
                throw new RuntimeException(e);
        }
}

private void displaydataset() {
        if (data.combineddata != null) { //check there is actually data to display
                try {
                        //get the graph settings the user wants
                        int index = graphselector.getSelectedIndex();
                        if (index < 0) index = 0; //sanity check
                        Graphsettings graphsettings = (Graphsettings)(graphselectorentries.get(index));
                        int decimatedsamplesperchannel = data.decimatedsamplesperchannel;
                        double [] xdata = new double[decimatedsamplesperchannel];
                        for (int i=0;i<decimatedsamplesperchannel;i++) {
                                xdata[i] = ((double)i)/10000;
                        }
                        Field field;

                        field = Data.class.getField(graphsettings.fieldname);
                        dataset.removeSeries(0);
                        double [] ydata;
                        if (graphsettings.channel <0) {
                                ydata = (double [])field.get(data);
```

```
                        } else {
                                ydata = ((double [][])field.get(data))[graphsettings.channel];
                        }
                        if (graphsettings.logm) {
                                double [] ydataabs = new double[decimatedsamplesperchannel];
                                for (int i=0;i<decimatedsamplesperchannel;i++) {
                                        ydataabs[i] = Math.abs(ydata[i]);
                                }
                                ydata = ydataabs;
                                setYAxisType(plot, LogarithmicAxis.class,graphsettings.yaxistitle);
                        } else {
                                setYAxisType(plot, NumberAxis.class,graphsettings.yaxistitle);
                        }
                        double [][] datakeyed = new double[2][];
                        datakeyed[0]=xdata;
                        datakeyed[1]=ydata;
                        dataset.addSeries(0, datakeyed);

                        chart.setTitle(graphsettings.toString());
                        descriptionlabel.setText(data.description);

                } catch (SecurityException e) {
                        throw new RuntimeException(e);
                } catch (NoSuchFieldException e) {
                        throw new RuntimeException(e);
                } catch (IllegalArgumentException e) {
                        throw new RuntimeException(e);
                } catch (IllegalAccessException e) {
                        throw new RuntimeException(e);
                }
        }
}

public static void main(String[] args) {

        try {
                new powermeasurementprocess(args);
        } catch (Exception e) {
                e.printStackTrace();
                return;
        }

}

public void actionPerformed(ActionEvent e) {
        String cmd = e.getActionCommand();
        if (cmd.equals("loaddatabutton")) {

                fc.setSelectedFile(new File(""));
                if (fc.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
                        try {
                                datafile = fc.getSelectedFile();
                                loadanddisplayfile(datafile);
                        } catch (Exception ex) {
                                ex.printStackTrace();
                                JOptionPane.showMessageDialog(this,"exception of type "+ex.getClass().getName()+" with message "+ex.getMessage());
                        }
                }

        }


        if (cmd.equals("exportbutton")) {
                //export to tab seperated text
                String filename = datafile.getAbsolutePath();
                filename = filename.substring(0, filename.length()-4) + ".txt";
                //String filename = "";

                //customised file chooser for export operation
                JFileChooser fcc = new JFileChooser();
                fcc.setSelectedFile(new File(filename));
                JPanel optionspanel = new JPanel();
                optionspanel.setLayout(new GridBagLayout());
                GridBagConstraints c = new GridBagConstraints();
                c.gridx = 0;
                c.gridwidth = 2;
                c.anchor = GridBagConstraints.WEST;
                JCheckBox filteredcheckbox = new JCheckBox("Export filtered data");
                c.gridy = 0;
                optionspanel.add(filteredcheckbox,c);
                JCheckBox scaledcheckbox = new JCheckBox("Export scaled data");
                c.gridy = 1;
                optionspanel.add(scaledcheckbox,c);
                JCheckBox combinedcheckbox = new JCheckBox("Export combined data",true);
                c.gridy = 2;
                optionspanel.add(combinedcheckbox,c);

                c.gridwidth = 1;
                c.gridx =1;
```

```
                c.gridy = 3;
                optionspanel.add(new JLabel("Start sample"),c);
                JTextField startsampletextfield = new JTextField("0",10);
                c.gridy = 4;
                optionspanel.add(startsampletextfield,c);
                c.gridy = 5;
                optionspanel.add(new JLabel("End sample"),c);
                JTextField endsampletextfield = new JTextField(""+(data.decimatedsamplesperchannel-1),10);
                c.gridy = 6;
                optionspanel.add(endsampletextfield,c);

                // add an empty panel to pad the text boxes and labels over a little
                JPanel padding = new JPanel();
                padding.setPreferredSize(new Dimension(10,0));
                c.gridx = 0;
                c.gridy = 3;
                c.gridheight = 4;
                optionspanel.add(padding,c);

                fcc.setAccessory(optionspanel);


                if (fcc.showSaveDialog(this) == JFileChooser.APPROVE_OPTION) {
                        try {
                                boolean dofiltered = filteredcheckbox.isSelected();
                                boolean doscaled = scaledcheckbox.isSelected();
                                boolean docombined = combinedcheckbox.isSelected();
                                int startsample = Integer.parseInt(startsampletextfield.getText());
                                int endsample = Integer.parseInt(endsampletextfield.getText());

                                PrintStream p = new PrintStream(fcc.getSelectedFile(),"WINDOWS-1252");

                                p.print("#"+data.description+"\r\n");
                                //print fixed part of header
                                p.print("#channel count: "+data.channelcount+"\r\n");
                                p.print("#data word size: "+data.datawordsize+"\r\n");
                                p.print("#samples per channel in filtered and decimated dataset"+data.decimatedsamplesperchannel+"\r\n");
                                p.printf("#time range in this file %d.%04d to %d.%04d\r\n",startsample/10000,startsample%10000,endsample/10000,endsample%10000);
                                p.print("#zero calibration voltage "+data.zerocalibrationvoltage+"\r\n");
                                //print per channel part of header
                                for (int channel = 0;channel<data.channelcount;channel++) {
                                        p.print("#channel "+channel+" zero calibration reading: "+data.zerocalibrationreadings[channel]+"\r\n");
                                        p.print("#channel "+channel+" loaded calibration reading: "+(int)data.loadedcalibrationreadings[channel]+"\r\n");
                                        p.print("#channel "+channel+" calibration resistor value: "+data.calibrationresistorvalues[channel]+"\r\n");
                                        p.print("#channel "+channel+" loaded calibration voltage: "+data.loadedcalibrationvoltages[channel]+"\r\n");
                                }
                                //print column headings
                                p.print("#time");
                                if (dofiltered) for (int channel = 0;channel<data.channelcount;channel++) {
                                        p.print("\tfiltered ch"+channel);
                                }
                                if (doscaled) for (int channel = 0;channel<data.channelcount;channel++) {
                                        p.print("\tscaled ch"+channel);
                                }
                                if (docombined) p.print("\tcombined");
                                p.print("\r\n");
                                //print data
                                for (int sample = startsample;sample <= endsample ;sample++) {
                                        p.printf("%d.%04d",sample/10000,sample%10000);
                                        if (dofiltered) for (int channel = 0;channel<data.channelcount;channel++) {
                                                //filtered but not scaled data is unlikely to have any significant
                                                //data after the decimal point, export with 2dp to make columns line up
                                                //nicely (any less and the titles end up spaced differently from the data)
                                                p.printf("\t%05.2f",data.filtereddata[channel][sample]);
                                        }
                                        if (doscaled) for (int channel = 0;channel<data.channelcount;channel++) {
                                                //format scaled data using computerised scientific notation
                                                //with 5 digits after the decimal point
                                                p.printf("\t%.5E",data.scaleddata[channel][sample]);
                                        }
                                        if (docombined) p.printf("\t%.5E",data.combineddata[sample]);
                                        p.print("\r\n");
                                }
                                p.close();

                        } catch (Exception ex) {
                                ex.printStackTrace();
                                JOptionPane.showMessageDialog(this,ex.getMessage(), "ERROR", JOptionPane.ERROR_MESSAGE);
                        }
                }

        }
        if (cmd.equals("exportsummarybutton")) {
                System.out.println("foo");
                int blockstart = 0;
                int blockcount = 1;

                int totalsamples = data.decimatedsamplesperchannel;
                //totalsamples = 1000;
```

```
String filename = datafile.getAbsolutePath();
filename = filename.substring(0, filename.length()-4) + "-summary.txt";
//String filename = "";

//customised file chooser for export operation
JFileChooser fcc = new JFileChooser();
fcc.setSelectedFile(new File(filename));
JPanel optionspanel = new JPanel();
optionspanel.setLayout(new GridBagLayout());
GridBagConstraints c = new GridBagConstraints();

c.gridwidth = 1;
c.gridx =1;
c.gridy = 0;
optionspanel.add(new JLabel("Averaging length"),c);
JTextField averaginglengthtextfield = new JTextField("50",10);
c.gridy = 1;
optionspanel.add(averaginglengthtextfield,c);
c.gridy = 2;
optionspanel.add(new JLabel("Exclude entries shorter than"),c);
JTextField minsizetextfield = new JTextField("10",10);
c.gridy = 3;
optionspanel.add(minsizetextfield,c);

// add an empty panel to pad the text boxes and labels over a little
JPanel padding = new JPanel();
padding.setPreferredSize(new Dimension(10,0));
c.gridx = 0;
c.gridy = 0;
c.gridheight = 4;
optionspanel.add(padding,c);

fcc.setAccessory(optionspanel);


if (fcc.showSaveDialog(this) == JFileChooser.APPROVE_OPTION) {
        try {

                int averaginglength = Integer.parseInt(averaginglengthtextfield.getText());
                int minsize = Integer.parseInt(minsizetextfield.getText());

                PrintStream p = new PrintStream(fcc.getSelectedFile(),"WINDOWS-1252");

                p.print("#"+data.description+"\r\n");
                //print fixed part of header
                p.print("#channel count: "+data.channelcount+"\r\n");
                p.print("#data word size: "+data.datawordsize+"\r\n");
                p.print("#samples per channel in filtered and decimated dataset"+data.decimatedsamplesperchannel+"\r\n");
                p.print("#zero calibration voltage "+data.zerocalibrationvoltage+"\r\n");
                //print per channel part of header
                for (int channel = 0;channel<data.channelcount;channel++) {
                        p.print("#channel "+channel+" zero calibration reading: "+data.zerocalibrationreadings[channel]+"\r\n");
                        p.print("#channel "+channel+" loaded calibration reading: "+(int)data.loadedcalibrationreadings[channel]+"\r\n");
                        p.print("#channel "+channel+" calibration resistor value: "+data.calibrationresistorvalues[channel]+"\r\n");
                        p.print("#channel "+channel+" loaded calibration voltage: "+data.loadedcalibrationvoltages[channel]+"\r\n");
                }
                //print column headings
                p.print("#start time\tend time\tsamples\taverage");
                p.print("\r\n");




                for (int sample=1;sample<totalsamples;sample++) {
                        //work out how many samples are already in current block
                        int samplesinblock = sample-blockstart;
                        //average sample already in current block or n samples starting at start of current block, whichever is longer
                        int samplestoaverage = Math.max(samplesinblock,averaginglength);
                        //make sure we don't overrun end of data set
                        if (blockstart+samplestoaverage > totalsamples) samplestoaverage = totalsamples-blockstart;
                        double blockaverage = 0;
                        for (int i=0;i<samplestoaverage;i++) {
                                blockaverage += data.combineddata[blockstart+i];
                        }
                        blockaverage = blockaverage / samplestoaverage;

                        //average n samples starting at current position
                        samplestoaverage = averaginglength;
                        //make sure we don't overrun end of data set
                        if (sample+samplestoaverage > totalsamples) samplestoaverage = totalsamples-sample;
                        double newaverage = 0;
                        for (int i=0;i<samplestoaverage;i++) {
                                newaverage += data.combineddata[sample+i];
                        }
                        newaverage = newaverage /samplestoaverage;
                        //check if new average is within 1% of value calculated above
                        //System.out.println("block average "+blockaverage+" new average "+newaverage);
                        double averagediv = newaverage/blockaverage;
```

```
                                        if ((averagediv > 0.95)&&(averagediv < 1.05)) {
                                                //do nothing, sample will enter current block
                                        } else {
                                                //we are going to start a new block

                                                //if the existing block is big enough to keep then write it out
                                                if (samplesinblock >= minsize) {

                                                        //if we previously averaged beyond the end of the block then reaverage only what is
                                                        //in the block
                                                        if (samplestoaverage != samplesinblock) {
                                                                samplestoaverage = samplesinblock;
                                                                blockaverage = 0;
                                                                for (int i=0;i<samplestoaverage;i++) {
                                                                        blockaverage += data.combineddata[blockstart+i];
                                                                }
                                                                blockaverage = blockaverage / samplestoaverage;
                                                        }

                                                        //output data for current block
                                                        p.printf("%d.%04d",blockstart/10000,blockstart%10000);
                                                        p.printf("\t%d.%04d",(sample-1)/10000,(sample-1)%10000);
                                                        p.print("\t"+samplesinblock);
                                                        p.println("\t"+blockaverage);
                                                }

                                                //begin new block with current sample
                                                blockstart = sample;
                                                blockcount++;
                                                System.out.println("starting new block at sample "+sample);
                                        }

                                }
                                System.out.println("data summarised into "+blockcount+" blocks");
                                p.close();

                        } catch (Exception ex) {
                                ex.printStackTrace();
                                JOptionPane.showMessageDialog(this,ex.getMessage(), "ERROR", JOptionPane.ERROR_MESSAGE);
                        }
                }

        }
}


private void loadanddisplayfile(File file) throws IOException, DataFormatException {
        loadfile(file,data);
        graphselectorentries.clear();

        int selectedindex =data.channelcount*3+1;
        System.out.println("populating list and selecting entry "+selectedindex);

        addlistentriesforchannels(graphselectorentries, "filtereddata", "filtered data",false,"adc reading","(ADC units)");
        addlistentriesforchannels(graphselectorentries, "scaleddata", "scaled data",true,"current","(uA)");
        graphselectorentries.addElement(new Graphsettings( "combineddata", -1, "combined data",false,"current (uA)"));
        graphselectorentries.addElement(new Graphsettings( "combineddata", -1, "combined data",true,"current magnitude (uA)"));

        graphselector.setSelectedIndex(selectedindex);

        displaydataset();
}

//reset buffers position to zero and read n bytes into it from the file, throw exception if number of bytes read
//is not exactly n
private static void readbuffer(FileInputStream f, ByteBuffer b, int n) throws IOException {
        b.position(0);
        b.limit(n);
        int r = f.getChannel().read(b);
        if (r != n) throw new IOException("wrong number of bytes read, expected "+n+" got "+r);
}

//note: char in java is an unsigned 16-bit number
private static void loadfile(File file, Data data) throws IOException, DataFormatException {
        Charset charset = Charset.forName("WINDOWS-1252");
        //read filter file
        int tapcount = 0;

        double [] filter = null;
        FileInputStream f = new FileInputStream("5khz 511 tap hanning filter.txt");
        try {

                for (int pass=0;pass<2;pass++) {
                        f.getChannel().position(0);
                        BufferedReader b = new BufferedReader(new InputStreamReader(f,charset));
                        String line = b.readLine();
                        int i=0;
                        while (line != null) {
```

```
                        if (pass == 0) {
                                //first pass: count lines
                                tapcount++;
                        } else {
                                //second pass read data
                                String [] tokens = line.trim().split(" +");
                                //System.out.println(tokens.length);
                                //System.out.println("!"+tokens[0]+"!");
                                //System.out.println("!"+tokens[1]+"!");
                                //second column contains the actual filter data
                                //filter generated by signal wizard has a DC gain of 4096, compensate for this
                                filter[i] = Double.parseDouble(tokens[1])/4096;

                                //System.out.println(filter[i]);
                        }
                        line = b.readLine();
                        i++;
                }
                if (pass == 0) {
                        //System.out.println("creating array for filter coefficients");
                        filter = new double[tapcount];

                }

        }

} finally {
        f.close();
}
System.out.println("filter has "+tapcount+" taps.");
displayminmax("filter coefficients", filter);
if ((tapcount % 2) != 1) throw new DataFormatException("tap count must be an odd number");
int centertap = tapcount /2;
byte channels;
double[] microampsperadcunit;
int[] zerocalibrationreadings;

//now read actual data file
f = new FileInputStream(file);
try {
        ByteBuffer hb = ByteBuffer.allocate(16); //header buffer
        hb.order(ByteOrder.LITTLE_ENDIAN);

        //read pre-header
        readbuffer(f,hb,16);
        String fileidentifier = new String(hb.array(),0,12,charset);
        if (!fileidentifier.equals("WSNPWRMEASUR")) throw new DataFormatException("failed to find file identification, did this file really come from the power measurement rig?!");
        char version = hb.getChar(12);
        System.out.println("version 0x"+Integer.toHexString(version));
        if ((version/256) > 0) throw new DataFormatException("version too high for this code to read");
        char headersize = hb.getChar(14);

        //read header from file
        hb = ByteBuffer.allocate(headersize);
    hb.order(ByteOrder.LITTLE_ENDIAN);
    readbuffer(f,hb,headersize);

    //read fixed part of header
    channels = hb.get(0);
    data.channelcount = channels;
    byte datawordsize = hb.get(1);
    data.datawordsize = datawordsize;
    char descoffset = hb.getChar(2);
    int totalsamplesperchannel = hb.getInt(4);
    double zerocalibrationvoltage = hb.getDouble(8);
    data.zerocalibrationvoltage = zerocalibrationvoltage;

    //read description from header
    System.out.println("description ofset 0x"+Integer.toHexString(descoffset));
    int desclen;
    {
            int i;
            //find null terminator on description
            for (i=descoffset;i<hb.capacity();i++) {
                    if (hb.get(i) == 0) break;
            }
            desclen = i - descoffset;
    }
    String description = new String(hb.array(),descoffset,desclen,charset);
    data.description = description;
    //read per channel part of header
    zerocalibrationreadings = new int[channels];
    char[] loadedcalibrationreadings = new char[channels];
    int[] calibrationresistorvalues = new int[channels];
    double[] loadedcalibrationvoltages = new double[channels];
    microampsperadcunit = new double[channels];
    for (int i = 0; i < channels; i++) {
            int cds = (i*16)+16;
            zerocalibrationreadings[i] = hb.getChar(cds);
```

```
        loadedcalibrationreadings[i] = hb.getChar(cds*2);
        calibrationresistorvalues[i] = hb.getInt(cds*4);
        loadedcalibrationvoltages[i] = hb.getDouble(cds*8);

        double microamps = loadedcalibrationvoltages[i]/calibrationresistorvalues[i]*1000000;
        microampsperadcunit[i] = microamps / (loadedcalibrationreadings[i]-zerocalibrationreadings[i]);
}

data.zerocalibrationreadings = zerocalibrationreadings;
data.loadedcalibrationreadings = loadedcalibrationreadings;
data.calibrationresistorvalues = calibrationresistorvalues;
data.loadedcalibrationvoltages = loadedcalibrationvoltages;

int decimationratio = 10;

//total samples (and block size) must be a multiple of decimation ratio for decimation to work correctly
if ((totalsamplesperchannel % decimationratio) != 0) throw new DataFormatException("total samples must be a multiple of "+decimationratio);

//initialise filtered data buffer
data.filtereddata = new double[3][totalsamplesperchannel/decimationratio];

//initialise scaled data buffer
data.scaleddata = new double[3][totalsamplesperchannel/decimationratio];

//initialise combined data buffer
data.combineddata = new double[totalsamplesperchannel/decimationratio];


//find a reasonable working block size that is a factor of total samples per channel
//and a multiple of the decimation ration
//start at 100000 which is the working block size used on the current rig
int blocksize = 10000 * decimationratio;
while ((totalsamplesperchannel % blocksize) != 0) blocksize +=decimationratio;
System.out.println("reading with block size "+blocksize);
int blockcount = totalsamplesperchannel / blocksize;

int decimatedblocksize = blocksize/decimationratio;
data.decimatedblocksize = decimatedblocksize;
data.decimatedsamplesperchannel = totalsamplesperchannel/decimationratio;


//prepare data structures need to read data from file and process it
if (datawordsize != 2) throw new DataFormatException("only 16-bit samples are currently supported");
char [] prevblock = new char[ blocksize*channels];
char [] currentblock = new char[blocksize*channels];
char [] nextblock = new char[blocksize*channels];

fillblockwithzerocalibrationvalues(channels,zerocalibrationreadings, blocksize, currentblock);

ByteBuffer db = ByteBuffer.allocate(blocksize*channels*datawordsize);
db.order(ByteOrder.LITTLE_ENDIAN);
CharBuffer dbs = db.asCharBuffer();

System.out.println("reading "+blockcount+" blocks of "+blocksize+ "sample points ("+blocksize*channels*datawordsize+" bytes)");


System.out.println("reading block 0");
readbuffer(f, db, blocksize*channels*datawordsize);
dbs.position(0);
dbs.get(nextblock);

//main processing loop
//note that each iteration of this loop works with three blocks of data
//the "previous block", the "current block" and the "next block".
//this is necessary because to produce a cleanly filtered result the
//filter algorithm must look ahead and behind into the previous and next blocks
//for the first iteration the "previous block" is filled with the zero calibration value
//and likewise for the last iteration the "next block" is filled with the zero calibration
//value
for (int blockno=0;blockno<blockcount;blockno++) {

        //rotate the buffers
        char [] tempblock = prevblock;
        prevblock = currentblock;
        currentblock = nextblock;
        nextblock = tempblock;

        //read NEXT block if available
        //otherwise fill NEXT block with zero calibration values
        if (blockno < (blockcount -1)) {
                System.out.println("reading block "+(blockno+1));
                readbuffer(f, db, blocksize*channels*datawordsize);
                dbs.position(0);
                dbs.get(nextblock);
        } else {
                //Arrays.fill(nextblock, (char)0);
                fillblockwithzerocalibrationvalues(channels,zerocalibrationreadings, blocksize, nextblock);

        }
```

```
                    System.out.println("processing block "+blockno);
                    displayminmax("block "+blockno,currentblock);
                    filteranddecimateblock(data, tapcount, filter, centertap,
                                    channels, decimationratio, blocksize, prevblock,
                                    currentblock, nextblock, blockno);

                    scaleblock(data, channels, zerocalibrationreadings,
                                    microampsperadcunit, decimatedblocksize,
                                    blockno);
                    if (channels != 3) throw new DataFormatException("current recombination algorithm only supports files with exactly 3 channels of data");
                    double[] scaledhdata = data.scaleddata[2];
                    double[] scaledmdata = data.scaleddata[1];
                    double[] scaledldata = data.scaleddata[0];
                    double[] cdata = data.combineddata;

                    for (int i=0;i<(decimatedblocksize );i++) {
                                int pos = (blockno * (decimatedblocksize))+i;
                            if (scaledhdata[pos] < 8) {
                                cdata[pos] = scaledhdata[pos];
                            } else if (scaledmdata[pos] < 800) {
                                cdata[pos] = scaledmdata[pos];
                            } else {
                                cdata[pos] = scaledldata[pos];
                            }

                    }

                }

            } finally{

                    f.close();
            }
            System.out.println("processing summary");
            for (int i=0;i<channels;i++) {
                    System.out.println("channel "+i+" zerocalibrationreading: "+zerocalibrationreadings[i]);
                    System.out.println("channel "+i+" microamps per adc unit: "+microampsperadcunit[i]);
                    displayminmax("channel "+i+" filtered and decimated",data.filtereddata[i]);
                    displayminmax("channel "+i+" scaled",data.scaleddata[i]);
            }
            displayminmax("combined data",data.combineddata);

    }

    private static void fillblockwithzerocalibrationvalues(byte channels,
                    int[] zerocalibrationreadings, int blocksize, char[] block) {
            for (int i =0 ; i < blocksize ; i++) {
                    for (int channel = 0; channel < channels; channel++) {
                            block[i*channels+channel] = (char)(zerocalibrationreadings[channel]);
                    }
            }
    }

    private static void displayminmax(String name, char[] array) {
            int min = array[0];
            int max = min;
            for (int i=1;i<array.length;i++) {
                    char value = array[i];
                    if (value < min) min = value;
                    if (value > max) max = value;
            }
            System.out.println(name+" min:"+min+" max:"+max);
    }

    private static void displayminmax(String name,double[] array) {
            double min = array[0];
            double max = min;
            for (int i=1;i<array.length;i++) {
                    double value = array[i];
                    if (value < min) min = value;
                    if (value > max) max = value;
            }
            System.out.println(name+" min:"+min+" max:"+max);
    }

    private static void scaleblock(Data data, byte channels,
                    int[] zerocalibrationreadings, double[] microampsperadcunit,
                    int decimatedblocksize, int blockno) {
            for (int channel =0;channel<channels;channel++) {
                    int zerocalibrationreading = zerocalibrationreadings[channel];
                    double microampsperadcunit_ = microampsperadcunit[channel];
                    double[] datain = data.filtereddata[channel];
                    double[] dataout = data.scaleddata[channel];
                    for (int i=0;i<(decimatedblocksize );i++) {
                            int pos = (blockno * (decimatedblocksize))+i;
                            dataout[pos] = (datain[pos]-zerocalibrationreading)*microampsperadcunit_;
                    }
            }
    }
```

```
	private static void filteranddecimateblock(Data data, int tapcount,
			double[] filter, int centertap, byte channels,
			int decimationratio, int blocksize, char[] prevblock,
			char[] currentblock, char[] nextblock, int blockno) throws DataFormatException {
		for (int channel =0;channel<channels;channel++) {

			double[] dataout = data.filtereddata[channel];
			for (int i=0;i<(blocksize/decimationratio);i++) {
				int inputcenter = i * 10;
				int outputpos = (blockno * (blocksize/decimationratio))+i;
				double outputvalue =0;

				if (((inputcenter-centertap) >= 0) && ((inputcenter+tapcount-1) < blocksize)) {
					//use simplified loop:
					for (int tap=0;tap<tapcount;tap++) {
						int inputpos = inputcenter + tap - centertap;
						int inputvalue = currentblock[(inputpos*channels)+channel];
						outputvalue += (inputvalue * filter[tap]);
					}
				} else {
					//use full loop
					for (int tap=0;tap<tapcount;tap++) {
						int inputpos = inputcenter + tap - centertap;
						//we may need to retrieve input value from either the previous block, the current
						//block or the next block
						int inputvalue;
						if (inputpos < 0) {
							//Retrieve from previous block
							inputvalue = prevblock[((inputpos+blocksize)*channels)+channel];
						} else if (inputpos >= blocksize) {
							//Retrieve from next block
							inputvalue = nextblock[((inputpos-blocksize)*channels)+channel];
						} else {
							//Retrieve from current block
							inputvalue = currentblock[(inputpos*channels)+channel];
						}
						double tapvalue = filter[tap];
						double outputcontribution = (inputvalue * tapvalue);
						outputvalue += outputcontribution;

					}
				}
				dataout[outputpos] = outputvalue;
				//throw new DataFormatException("tapcount = "+tapcount+" outputvalue = "+outputvalue);


			}
		}

	}

	public void valueChanged(ListSelectionEvent arg0) {
		displaydataset();
	}
}
```

Appendix H

Field measurement control source code

This appendix is a place holder for the code used to control the field propagation measurement system described in Chapter 6. It was based on example code provided by Anritsu. The comments in the Anritsu sample code indicated that it was in turn based on example code provided by National Instruments. National Instruments visa is used to control the instruments.

The actual code is not presented in the thesis itself due to copyright concerns but is available to the examiners on the included disc in the directory “fieldmeasurement”.

Appendix I

PA measurement control source code

This appendix is a place holder for the code used to control the power amplifier gain and harmonic measurement system described in Chapter 6. It was based on the control source for the field propagation measurement system which was in turn based on example code provided by Anritsu. The comments in the Anritsu sample code indicated that it was in turn based on example code provided by National Instruments. The program is written in C++. NI visa is used to control the instruments.

The actual code is not presented in the thesis itself due to copyright concerns but is available to the examiners on the included disc in the directory “ampmeasurement”.

Appendix J

RF measurement processing source code

This appendix contains the source code used to process the results of the two RF measurement systems described in chapter 6. The code was written from scratch by the author in Java. JFreechart is used for plotting. The source code is presented below and can also be found in the file fieldmeasurementprocess.java in the "fieldmeasurementprocess" directory on the disc included for the examiners.

```
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.nio.ByteBuffer;
import java.nio.ByteOrder;
import java.nio.FloatBuffer;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.zip.DataFormatException;

import javax.imageio.IIOException;
import javax.swing.*;
import javax.swing.event.ListDataListener;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;

import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.chart.plot.XYPlot;
import org.jfree.data.xy.DefaultXYDataset;

class Measurement {

        public Measurement(int transmitfreq, int samplecount, int power,int harmonic, int span, float[] samples, float max) {
                this.transmitfreq = transmitfreq;
                this.samplecount = samplecount;
                this.power = power;
                this.harmonic = harmonic;
                this.span = span;
                this.samples = samples;
                this.max = max;
        }
        int transmitfreq;
        int samplecount;
        int power;
        int harmonic;
        int span;

        float [] samples;
        float max;
}
```

```
//list model with duplicate elimination
//based on a defaultlistmodel and two hashmaps
//each element has a key, bidirectional mapping is provided between keys and
//list entries.
//currently it is only possible to clear the list and add to the list not to remove
//entries
class NoRepeatsListModel implements ListModel {
        DefaultListModel wrappedmodel = new DefaultListModel();
        HashMap<Integer, Integer> forward = new HashMap<Integer, Integer>();
        HashMap<Integer, Integer> reverse = new HashMap<Integer, Integer>();

        // add an element if it doesn't already exist and return it's position
        public int addElement(String s, int key) {
                int position;
                if (forward.containsKey(key)) {
                        position = forward.get(key);
                } else {
                        wrappedmodel.addElement(s);
                        position = wrappedmodel.size()-1;
                        forward.put(key, position);
                        reverse.put(position, key);
                }
                return position;
        }

        public void clear() {
                wrappedmodel.clear();
                forward.clear();
                reverse.clear();
        }

        //pass calls from listmodel interface to wrapped listmodel
        public void addListDataListener(ListDataListener l) {
                wrappedmodel.addListDataListener(l);
        }
        public Object getElementAt(int index) {
                return wrappedmodel.getElementAt(index);
        }
        public int getSize() {
                return wrappedmodel.getSize();
        }
        public void removeListDataListener(ListDataListener l) {
                wrappedmodel.removeListDataListener(l);
        }

}


@SuppressWarnings("serial")
public class fieldmeasurementprocess extends JFrame implements ListSelectionListener, ActionListener {

        /**
         * @param args
         */

        private ArrayList<Measurement> data = new ArrayList<Measurement>();
        private ArrayList<Measurement> caldata = new ArrayList<Measurement>();
        private boolean havecaldata = false;

        private DefaultXYDataset dataset;
        private JList freqlist;
        private JFreeChart chart;
        private NoRepeatsListModel freqlistentries;

        private int currentdisplaypower = -3000000;
        private int currentdisplayfrequency = -3000000;
        private int currentdisplayharmonic = -3000000;

        private XYPlot plot;
        private double datahighrounded=-10;
        private double datalowrounded=10;

        private JFileChooser fc = new JFileChooser("d:\\");
        private File datafile;
        private File calfile;
        private NoRepeatsListModel powerlistentries;
        private JList powerlist;
        private NoRepeatsListModel harmoniclistentries;
        private JList harmoniclist;


        public fieldmeasurementprocess(String[] args) {


                //displayfile("Y:||fieldmeasurementrecordings||fieldmeasurement_20110124corridoortest.dat");

                setSize(500, 500);
                dataset = new DefaultXYDataset();
```

```
        displaydataset(-3000000,-3000000,-3000000, false);

        chart = ChartFactory.createXYLineChart("no data loaded","frequency (Mhz)","level (dbm)",dataset,PlotOrientation.VERTICAL,false,false,false);
        plot = (XYPlot) chart.getPlot();
        //setsummarychartrange(data.size());
        //plot.getRangeAxis().setRange(datalowrounded,datahighrounded);
        ChartPanel chartp = new ChartPanel(chart);
        add(chartp);

        JPanel leftpanel = new JPanel(new BorderLayout());
        add(leftpanel,BorderLayout.WEST);

        powerlistentries = new NoRepeatsListModel();
        powerlistentries.addElement("no data loaded",-3000000);
        powerlist = new JList(powerlistentries);
        powerlist.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        powerlist.setSelectedIndex(0);
        powerlist.addListSelectionListener(this);
        JScrollPane powerlistscrollpane = new JScrollPane(powerlist);
        leftpanel.add(powerlistscrollpane,BorderLayout.WEST);


        freqlistentries = new NoRepeatsListModel();
        freqlistentries.addElement("no data loaded",-3000000);
        freqlist = new JList(freqlistentries);
        freqlist.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        freqlist.setSelectedIndex(0);
        freqlist.addListSelectionListener(this);
        JScrollPane freqlistscrollpane = new JScrollPane(freqlist);
        leftpanel.add(freqlistscrollpane,BorderLayout.CENTER);


        harmoniclistentries = new NoRepeatsListModel();
        harmoniclistentries.addElement("no data loaded",-3000000);
        harmoniclist = new JList(harmoniclistentries);
        harmoniclist.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        harmoniclist.setSelectedIndex(0);
        harmoniclist.addListSelectionListener(this);
        JScrollPane harmoniclistscrollpane = new JScrollPane(harmoniclist);
        leftpanel.add(harmoniclistscrollpane,BorderLayout.EAST);

        JPanel buttonbar = new JPanel();
        add(buttonbar,BorderLayout.SOUTH);


        JButton loaddatabutton = new JButton("load data");
        loaddatabutton.setActionCommand("loaddatabutton");
        loaddatabutton.addActionListener(this);
        buttonbar.add(loaddatabutton);

        JButton loadcalbutton = new JButton("load calibration");
        loadcalbutton.setActionCommand("loadcalbutton");
        loadcalbutton.addActionListener(this);
        buttonbar.add(loadcalbutton);

        JButton mergebutton = new JButton("merge data");
        mergebutton.setActionCommand("mergebutton");
        mergebutton.addActionListener(this);
        buttonbar.add(mergebutton);

        JButton savebutton = new JButton("save data");
        savebutton.setActionCommand("savebutton");
        savebutton.addActionListener(this);
        buttonbar.add(savebutton);

        JButton exportbutton = new JButton("export");
        exportbutton.setActionCommand("exportbutton");
        exportbutton.addActionListener(this);
        buttonbar.add(exportbutton);
        setDefaultCloseOperation(DISPOSE_ON_CLOSE);
        setVisible(true);
}

public static void main(String[] args) {

        try {
                new fieldmeasurementprocess(args);
        } catch (Exception e) {
                e.printStackTrace();
                return;
        }

}

//reset buffers position to zero and read n bytes into it from the file, throw exception if number of bytes read
//is not exactly n
private static void readbuffer(FileInputStream f, ByteBuffer b, int n) throws IOException {
        b.position(0);
```

```
        b.limit(n);
        int r = f.getChannel().read(b);
        if (r != n) throw new IIOException("wrong number of bytes read");
}



private static void loadfile(File file,ArrayList<Measurement> data) throws IOException, DataFormatException {
        data.clear();
        FileInputStream f = new FileInputStream(file);

        int dbs = 0; //data buffer size
        ByteBuffer db = null;// data buffer
        FloatBuffer dbf = null; //data buffer as float buffer

        ByteBuffer hb = ByteBuffer.allocate(16); //header buffer
        hb.order(ByteOrder.LITTLE_ENDIAN);

        while (f.getChannel().size() > f.getChannel().position()) {

                readbuffer(f,hb,4);
                int version = hb.getInt(0);
                int transmitfreq; //transmit frequency in kHz
                int power; //nominal transmit power in hundreths of a dBm
                int harmonic; //receive harmonic (value by which transmit frequency is multiplied to get receive center frequency;
                int span; //receiver frequency span in Hz

                if (version < 0x01000000) {
                        // old format, pre introduction of versioning
                        transmitfreq = version;
                        power = 0;
                        harmonic = 1;
                        span = 2000;
                } else {
                        readbuffer(f,hb,16);
                        transmitfreq = hb.getInt(0);
                        power = hb.getInt(4);
                        harmonic = hb.getInt(8);
                        span = hb.getInt(12);
                }
                readbuffer(f,hb,4);
                int databytes = hb.getInt(0);
                if ((databytes %4) != 0) throw new DataFormatException("number of data bytes must be a multiple of 4");
                int samplecount = databytes / 4;
                if (samplecount <= 0) throw new DataFormatException("sample count must be greater than zero");

                //System.out.printf("center frequency %d, samples %d\n", centerfreq,samplecount);
                if (dbs < databytes) { //if the existing buffer is not big enough (or doesn't exist yet) replace it with a new one
                        // in most cases all the recordings in a file will be the same size so this should be pretty efficiant in practice

                        db = ByteBuffer.allocate(databytes);
                        db.order(ByteOrder.LITTLE_ENDIAN);
                        dbf        = db.asFloatBuffer();
                }
                readbuffer(f, db, databytes);
                dbf.position(0);
                float samples[] = new float[samplecount];
                dbf.get(samples);

                float max = samples[0];
                for (int i=1;i<samplecount;i++) {
                        if (samples[i] > max) max = samples[i];
                }
                data.add(new Measurement(transmitfreq,samplecount,power,harmonic, span, samples, max));

        }

        f.close();
}

private static void savefile(ArrayList<Measurement> data, File file) throws IOException {
        System.out.println("in savefile");
        FileOutputStream f = new FileOutputStream(file);

        int dbs = 0; //data buffer size
        ByteBuffer db = null;// data buffer
        FloatBuffer dbf = null; //data buffer as float buffer

        ByteBuffer hb = ByteBuffer.allocate(24); //header buffer
        hb.order(ByteOrder.LITTLE_ENDIAN);

        int datasize = data.size();
        for (int i=0;i<datasize;i++) {
                System.out.println("saving data item "+i);
                Measurement dataitem = data.get(i);

                int version = 0x01000000;
                hb.putInt(0,version);
                hb.putInt(4,dataitem.transmitfreq);
```

```
                hb.putInt(8,dataitem.power);
                hb.putInt(12,dataitem.harmonic);
                hb.putInt(16,dataitem.span);
                int databytes = dataitem.samplecount *4;
                hb.putInt(20,databytes);
                hb.position(0);
                f.getChannel().write(hb);

                //System.out.printf("center frequency %d, samples %d\n", centerfreq,samplecount);
                if (dbs < databytes) { //if the existing buffer is not big enough (or doesn't exist yet) replace it with a new one
                        // in most cases all the recordings in a file will be the same size so this should be pretty efficiant in practice

                        db = ByteBuffer.allocate(databytes);
                        db.order(ByteOrder.LITTLE_ENDIAN);
                        dbf        = db.asFloatBuffer();
                }

                dbf.position(0);
                float samples[] = dataitem.samples;
                dbf.put(samples);
                db.position(0);
                db.limit(databytes);
                f.getChannel().write(db);

        }

        f.close();

}

private void handlemerge(File file) throws IOException, DataFormatException {
        ArrayList<Measurement> loadeddata = new ArrayList<Measurement>();

        loadfile(file,loadeddata);

        @SuppressWarnings("unchecked") //data.clone should be the same type as
                                       //data
        ArrayList<Measurement> newdata = (ArrayList<Measurement>)data.clone();

        int loadeddatasize = loadeddata.size();

        for (int i=0;i<loadeddatasize;i++) {
                Measurement loadeddataitem = loadeddata.get(i);
                //look for matching or preceding element in existing data
                int foundelement = -1;
                boolean perfectmatch = false;
                int newdatasize = newdata.size();
                for (int j=0; j<newdatasize;j++) {
                        Measurement dataitem = newdata.get(j);
                        if (dataitem.transmitfreq < loadeddataitem.transmitfreq) {
                                foundelement = j;
                        } else if (dataitem.transmitfreq == loadeddataitem.transmitfreq) {
                                if (dataitem.power < loadeddataitem.power) {
                                        foundelement = j;
                                } else if (dataitem.power == loadeddataitem.power) {
                                        if (dataitem.harmonic < loadeddataitem.harmonic) {
                                                foundelement = j;
                                        } else if (dataitem.harmonic == loadeddataitem.harmonic) {
                                                foundelement = j;
                                                perfectmatch = true;
                                                break;
                                        }
                                }
                        }
                }

                if (perfectmatch) {
                        // we have found a perfect match, we need to ask the user about merge options
                        int option = JOptionPane.showOptionDialog(this,
                          "the file being merged and the data in memory both have recordings at freq "+loadeddataitem.transmitfreq+
                          ",power "+loadeddataitem.power+"and harmonic "+loadeddataitem.harmonic, "overlap handling",
                          0, JOptionPane.QUESTION_MESSAGE, null, new String[] {"keep","overwrite","cancel"}, null);

                        switch (option){
                                case 0: //keep
                                        //do nothing
                                break;
                                case 1: //overwrite
                                        //replace the measurement in memory with the loaded one
                                        newdata.set(foundelement, loadeddataitem);
                                break;
                                default:
                                        //the user either closed the dialog or clicked cancel, abort the merge
                                        return;
                        }
                } else {
                        //no existing point in dataset, perform insertion after foundelement
                        newdata.add(foundelement+1, loadeddataitem);
                }
```

```
        }
        //success
        data = newdata;
        fillllists();
}

private void loadanddisplayfile(File file) throws IOException, DataFormatException {
        loadfile(file,data);
        fillllists();
}


public static void resetListAndAddSummaryEntries(NoRepeatsListModel listentries , boolean havecaldata) {

        listentries.clear();
        //-1000000 and smaller numbers should never appear as a data value in any of the lists
        //since negative frequency is not pysically possible, as are negative harmonics
        //and -10000 dBm is far too low to be generated or detected by any equipment that exists
        //now or in the forseeable future.
        if (havecaldata) {
                listentries.addElement("sumary (uncalibrated)",-2000000);
                listentries.addElement("sumary (calibrated)",-1000000);
        } else {
                listentries.addElement("sumary",-2000000);
        }
}

//called after something that may impact the lists has changed for example new data or new cal data
private void fillllists() {
        resetListAndAddSummaryEntries(powerlistentries, havecaldata);
        resetListAndAddSummaryEntries(freqlistentries, havecaldata);
        resetListAndAddSummaryEntries(harmoniclistentries, havecaldata);

        int datasize = data.size();
        for (int i=0;i<datasize;i++) {
                Measurement dataitem = data.get(i);
                powerlistentries.addElement(""+dataitem.power,dataitem.power);
                freqlistentries.addElement(""+dataitem.transmitfreq,dataitem.transmitfreq);
                harmoniclistentries.addElement(""+dataitem.harmonic,dataitem.harmonic);
        }
        freqlist.invalidate();
        this.validate();
        int firstnonsummaryentry =1;
        if (havecaldata) firstnonsummaryentry = 2;
        powerlist.setSelectedIndex(firstnonsummaryentry);
        freqlist.setSelectedIndex(0);
        harmoniclist.setSelectedIndex(firstnonsummaryentry);
        valueChanged(null); //force redraw with new data
}


static class ProcessedData {
        String title = "";
        String xlabel ="";
        double [] xdata = new double[0];
        double xmin = -10;
        double xmax = 10;

        String ylabel ="";
        double [] ydata = new double[0];
        double ymin = -10;
        double ymax = 10;

        //does this graph represent real data or an error state?
        boolean real = false;
}

private static ProcessedData processdata(ArrayList<Measurement> data, ArrayList<Measurement> caldata, int power, int frequency, int harmonic) {
        ProcessedData result = new ProcessedData();
        if ((power == -3000000) || (frequency == -3000000) || (harmonic == -3000000)) {
                if (data.size() == 0) {
                        result.title = "no data loaded";
                } else {
                        result.title = "no selection";
                }
                return result;
        }
        double datad[];
        double datakeys[];
        boolean xaxispower = (power <= -1000000);
        boolean xaxisfrequency = (frequency <= -1000000);
        boolean xaxisharmonic = (harmonic <= -1000000);

        if ((xaxispower && xaxisfrequency)||(xaxispower && xaxisharmonic)||(xaxisfrequency && xaxisharmonic)) {
                result.title = "summary can only be selected on one list at a time";
                return result;
        }
```

```
boolean usingcal = false;
boolean summary = false;
result.title = "";
if (xaxisfrequency) {
        summary = true;
        usingcal = (frequency == -1000000);
} else {
   result.title += frequency+"kHz ";
}
if (xaxispower) {
        summary = true;
        usingcal = (power == -1000000);
} else {
        result.title += (((double)power)/100)+"dBm ";
}

if (xaxisharmonic) {
        summary = true;
        usingcal = (harmonic == -1000000);
} else {
        if (harmonic == 1) {
                result.title += "fundamental";
        } else if (harmonic == 2) {
                result.title += "2nd harmonic";
        } else if (harmonic == 2) {
                result.title += "3rd harmonic";
        } else {
                result.title += harmonic+"th harmonic";
        }
}
double datahigh,datalow;
result.title = result.title.trim();




if (summary) {
        int datasize = data.size();

        if (usingcal) {
          result.title += " (calibrated)";
        } else {
          result.title += " (uncalibrated)";
        }
        double xlow = 0;
        double xhigh = 0;
        datalow = 0;
        datahigh = 10;
        int dataptr = 0;
        datad = new double[datasize];
        datakeys = new double[datasize];
        if (datasize != 0) {

                for (int i=0;i<datasize;i++) {
                        Measurement dataitem = data.get(i);
                        if ((xaxispower || (power == dataitem.power))&&
                          (xaxisfrequency || (frequency == dataitem.transmitfreq))&&
                          (xaxisharmonic || (harmonic == dataitem.harmonic))) {
                                System.out.println("processing dataitem with power "+dataitem.power+
                                  " frequency "+dataitem.transmitfreq+" and harmonic"+dataitem.harmonic);
                                double key = 0;
                                int itempower = dataitem.power;
                                int itemfreq = dataitem.transmitfreq;
                                if (xaxispower) key = ((double)itempower)/100;
                                if (xaxisfrequency) key = ((double)itemfreq)/1000;
                                if (xaxisharmonic) key = dataitem.harmonic;
                                if (dataptr == 0) {
                                        xlow=xhigh=key;
                                } else {
                                        if (key < xlow) xlow=key;
                                        if (key > xhigh) xhigh=key;
                                }
                                double datavalue = dataitem.max;

                                if (usingcal) datavalue = applycaltovalue(caldata,datavalue, itemfreq, itempower);

                                datad[dataptr]=datavalue;
                                datakeys[dataptr] = key;
                                if (dataptr == 0) {
                                        datalow=datahigh=datavalue;
                                } else {
                                        if (datavalue > datahigh) datahigh=datavalue;
                                        if (datavalue < datalow) datalow=datavalue;
                                }
                                System.out.println("processed dataitem xlow="+xlow+" xhigh="+xhigh);
                                dataptr++;
                        }
                }
```

```
                }
                //truncate datad and datakeys arrays
                datad = Arrays.copyOf(datad, dataptr);
                datakeys = Arrays.copyOf(datakeys, dataptr);

                //jfreechart doesn't like low=high so add some margin if that happens to prevent errors
                if (xlow == xhigh) {
                        xlow -= 10;
                        xhigh +=10;
                }

                if (dataptr == 0) {
                        //no data to display, set some arbitary limits to avoid errors
                        xlow = 0;
                        xhigh = 1000;
                }
                System.out.println("xlow="+xlow+" xhigh="+xhigh);
                result.xmin = xlow;
                result.xmax = xhigh;

                if (xaxispower) result.xlabel = "transmit power (dBm)";
                if (xaxisfrequency) result.xlabel = "frequency (Mhz)";
                if (xaxisharmonic) result.xlabel = "harmonic";



        } else {
                Measurement dataitem = null;
                int datasize = data.size();
                for (int i=0;i<datasize;i++) {
                        Measurement potentialdataitem = data.get(i);
                        if ((power == potentialdataitem.power) && (frequency == potentialdataitem.transmitfreq) && (harmonic == potentialdataitem.harmonic)) {
                                dataitem = potentialdataitem;
                                break;
                        }

                }
                if (dataitem == null) {
                        result.title = "no data for this combination";
                        return result;
                }

                int samplecount = dataitem.samplecount;
                //int centerfreq = dataitem.transmitfreq;
                float [] samples = dataitem.samples;


                result.xmax = (((double)dataitem.span)/2/1000);
                result.xmin = -result.xmax;
                result.xlabel = "relative frequency (kHz)";
                datad = new double[samplecount];
                datakeys = new double[samplecount];
                int centersample = samplecount/2; // the below assumes an odd number of samples (which is typical for specan
                                                  // traces
                datahigh = datalow=samples[0];
                for (int i=0;i<samplecount;i++) {
                        //System.out.printf("%d: %f\n",i,samples[i]);
                        datakeys[i] = ((double)(centersample - i))/(samplecount-1)* dataitem.span / 1000 ;
                        double datavalue = samples[i];
                        datad[i] =datavalue;
                        if (datavalue > datahigh) datahigh=datavalue;
                        if (datavalue < datalow) datalow=datavalue;
                }
        }
        result.ymin = datalow;
        result.ymax = datahigh;
        result.ydata = datad;
        result.xdata = datakeys;
        if (usingcal) {
                result.ylabel = "propogation (db)";
        } else {
                result.ylabel = "level (dbm)";
        }
        result.real = true;
        return result;
}



private void displaydataset(int power, int frequency, int harmonic, boolean force) {

        if ((power != currentdisplaypower) || (frequency != currentdisplayfrequency) || (harmonic != currentdisplayharmonic) || force) {


                currentdisplaypower = power;
                currentdisplayfrequency = frequency;
                currentdisplayharmonic = harmonic;
                ProcessedData result = processdata(data, caldata, power, frequency, harmonic);
```

```
                datahighrounded = Math.ceil(result.ymax/10)*10;
                datalowrounded = Math.floor(result.ymin/10)*10;
                //avoid making a chart with zero vertical size
                if (datahighrounded == datalowrounded) {
                        datahighrounded += 10;
                        datalowrounded -= 10;
                }
                if (chart != null) { //test required because this can get called before the chart is fully created
                        chart.setTitle(result.title);
                        plot.getDomainAxis().setRange(result.xmin,result.xmax);
                        plot.getDomainAxis().setLabel(result.xlabel);
                        plot.getRangeAxis().setRange(datalowrounded,datahighrounded);
                        plot.getRangeAxis().setLabel(result.ylabel);

                }
                double [][] datakeyed = new double[2][];
                datakeyed[0]=result.xdata;
                datakeyed[1]=result.ydata;
                dataset.addSeries(0, datakeyed);
        }
        System.out.println(" datalowrounded="+datalowrounded+" datahighrounded="+datahighrounded);
}

private static double applycaltovalue(ArrayList<Measurement> caldata, double datavalue, int freq, int power) {
        //find a calibration entry with the same transmit power and frequency and a harmonic of 1
        int calindex = 0;
        int calentrycount = caldata.size();
        Measurement calentry = null;

        while (calindex < calentrycount) {
                calentry = caldata.get(calindex);
                if ((power == calentry.power) && (freq == calentry.transmitfreq) && (1 == calentry.harmonic)) break;
                calindex++;
        }

        if (calindex >= calentrycount) throw new IndexOutOfBoundsException(
          "unable to find appropriate entry in calibration data for frequency "+freq+" and power "+power);

        double calvalue = calentry.max;

        /* old interpolation code, not currently used {
                //use linear interpolation to find a cal value
                /*int calindexlow = calindexhigh - 1;

                int calfreqlow = caldata.get(calindexlow).transmitfreq;
                int calfreqhigh = caldata.get(calindexhigh).transmitfreq;
                int calfreqwidth = calfreqhigh - calfreqlow;
                double proportion = ((double)(freq-calfreqlow)) / calfreqwidth;
                calvalue = (caldata.get(calindexlow).max * (1-proportion))+(caldata.get(calindexhigh).max * (proportion));

        } */
        datavalue -= calvalue;

        return datavalue;
}

private int getlistvalue(JList list, NoRepeatsListModel listentries, int failure) {
        int result;
        int index = list.getSelectedIndex();
        if (index < 0) {
                result = failure;
        } else {
                result = listentries.reverse.get(index);
        }
        return result;
}

//value changed in the list
public void valueChanged(ListSelectionEvent e) {
        // if e is null then that means we were called from the load and display method
        // and as such want to force a refresh;
        boolean force = (e == null);

        int power = getlistvalue(powerlist, powerlistentries, -3000000);
        int frequency =  getlistvalue(freqlist,freqlistentries, -3000000);
        int harmonic =  getlistvalue(harmoniclist,harmoniclistentries, -3000000);

        displaydataset(power, frequency, harmonic, force);
}



public void actionPerformed(ActionEvent e) {
        String cmd = e.getActionCommand();
        if (cmd.equals("loaddatabutton")) {

                fc.setSelectedFile(new File(""));
                if (fc.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
                        try {
```

```
                        datafile = fc.getSelectedFile();
                        loadanddisplayfile(datafile);
                } catch (Exception ex) {
                        ex.printStackTrace();
                        JOptionPane.showMessageDialog(this,"exception of type "+ex.getClass().getName()+" with message "+ex.getMessage());
                }
        }

}
if (cmd.equals("mergebutton")) {

        fc.setSelectedFile(new File(""));
        if (fc.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
                try {
                        datafile = fc.getSelectedFile();
                        handlemerge(datafile);
                } catch (Exception ex) {
                        ex.printStackTrace();
                        JOptionPane.showMessageDialog(this,"exception of type "+ex.getClass().getName()+" with message "+ex.getMessage());
                }
        }

}
if (cmd.equals("savebutton")) {

        fc.setSelectedFile(new File(""));
        if (fc.showSaveDialog(this) == JFileChooser.APPROVE_OPTION) {
                try {
                        datafile = fc.getSelectedFile();
                        savefile(data,datafile);
                } catch (Exception ex) {
                        ex.printStackTrace();
                        JOptionPane.showMessageDialog(this,"exception of type "+ex.getClass().getName()+" with message "+ex.getMessage());
                }
        }

}

if (cmd.equals("loadcalbutton")) {

        fc.setSelectedFile(new File(""));
        if (fc.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
                try {
                        calfile = fc.getSelectedFile();
                        loadfile(calfile,caldata);
                        havecaldata = true;
                        filllists();
                } catch (Exception ex) {
                        JOptionPane.showMessageDialog(this,ex.getMessage());
                }
        }

}
if (cmd.equals("exportbutton")) {

        String filename = datafile.getAbsolutePath();
        filename = filename.substring(0, filename.length()-4) + ".csv";
        fc.setSelectedFile(new File(filename));

        if (fc.showSaveDialog(this) == JFileChooser.APPROVE_OPTION) {
                try {



                        int power = getlistvalue(powerlist, powerlistentries, -3000000);
                        int frequency =  getlistvalue(freqlist,freqlistentries, -3000000);
                        int harmonic =  getlistvalue(harmoniclist,harmoniclistentries, -3000000);
                        ProcessedData result = processdata(data, caldata, power, frequency, harmonic);
                        if (result.real == false) {

                                JOptionPane.showMessageDialog(this, result.title , "ERROR", JOptionPane.ERROR_MESSAGE);

                                return;
                        }
                        PrintStream p = new PrintStream(fc.getSelectedFile());
                        p.print(result.title+"\r\n");
                        p.print(result.xlabel+","+result.ylabel+"\r\n");
                        int processeddatasize = result.xdata.length;
                        for (int i=0;i<processeddatasize;i++) {
                                p.print(result.xdata[i]+","+result.ydata[i]+"\r\n");
                        }

                        p.close();

                } catch (Exception ex) {
                        JOptionPane.showMessageDialog(this,ex.getMessage(), "ERROR", JOptionPane.ERROR_MESSAGE);
                }
        }
```

```
                }
            }
        }
```

Appendix K

Source code for software running on sensor node hardware

This appendix is a place holder for the code that runs on the PIC micro controllers located in the sensor node hardware used in the project. It was based on Microchip's MiWi stack and associated demo applications. To achieve the author's goals modifications had to be made both to the demo applications and to the MiWi stack itself. The actual code is not presented in the thesis itself due to copyright concerns but is available to the examiners on the included disc in the directory "miwi".

The "miwi" directory has two subdirectories, "MiWi" and "Microchip". The "miwi/Microchip" directory contains the Microchip application libraries including the MiWi stack. As mentioned previously the MiWi stack has been modified for use in the author's project. A file "Microchip.diff" is included to show the modifications the author made to these files.

The "miwi/MiWi" directory contains the application software. There are five subdirectories, "MiWi Coordinator" and "MiWi End Device" contain the demonstration applications from Microchip that the author used as a base while "basestation", "repeater" and "node" contain the applications that were placed on the PICs in the Base station, repeater node and end nodes respectively.

The end node software was based on the "MiWi End Device" demonstration application while the base station and repeater software were based on the "MiWi Coordinator" demonstration application. Diff files have been included to show the changes the author made to the demonstration applications.

There are also three source files placed directly in the "miwi/MiWi" directory. These source files were created by the author and are shared between the base station, repeater and end node applications. The file "networksettings.h" contains the constant that determines how often the nodes wake up, make measurements and send a message to the base station. The file "rtcprocs.c" contains the code used to access the real time clock and "rtcprocs.h" provides the prototypes for those functions.

Appendix L

Source code for Linux gateway application in base station

This appendix contains the source code for the gateway application that runs on the Linux board in the base station. It was written from scratch in Free Pascal by the author. For serial communications it uses a library called lcore which was partially written by the author but not as part of this project, the intention was also to use this library for networking but in the end no network functionality was included in the application.

The source code is presented below and can also be found in the file gateway.dpr in the "gateway" directory on the disc included for the examiners.

```
program gateway;
uses
  math,
  sysutils,
  lcore,
  lcoreselect,
  lserial,
  btime,
  ctypes,
  baseunix,
  termio;

function makehexstring(s : string) : string;
var
  i : integer;
begin
  result := '';
  for i := 1 to length(s) do begin
    result := result + inttohex(ord(s[i]),2);
  end;
end;

type
  tsc=class
    procedure serialdataavailable(sender: tobject;error : word);
  end;

var
  lserial : tlserial;
  sc: tsc;
  buf: string;

const
  maxpartcount = 16;
type
  tparts = record
    parts : array[0..maxpartcount-1] of string;
        partcount : integer;
  end;
```

```
function breakupline(line: string) : tparts;
var
  partstart, partend, partlen,currentpart : integer;
  i : integer;

begin
  currentpart := 0;
  partstart := 1;
  result.partcount := 0;
  for i := 1 to length(line) do begin
    if (i=length(line)) or ((line[i] = ' ') and (currentpart < maxpartcount-1)) then begin
          partend := i-1;
          if (partend >= partstart) then begin //if partend is less than partstart we have a zero length part which we will ignore
            partlen := partend-partstart+1;
                result.parts[result.partcount] := copy(line,partstart,partlen);
                result.partcount := result.partcount + 1;
          end;
          partstart := i+1;
        end;
  end;
end;

//convert a byte to a 2 digit hexadecimal number
function hb(b : byte) : string; inline;
begin
  result := inttohex(b,2);
end;

//convert a byte to a 2 digit decimal number (used in timekeeping since the node rtc is bcd);
//note: numbers 100 and above will be converted to strings with non-digits as the first character
function d2d(b : byte) : string; inline;
begin
  result := char(byte('0')+ b div 10)+char(byte('0')+ b mod 10);
end;


procedure processserialline(line:string);
var
  parts : tparts;
  code : integer;
  t : extended;
  y,m,d,hr,min,sec,ms : word;
  //sourcelongaddress : array [0..7] of byte;
  f : file;
  logfilename : string;
  logline : string;
begin
  parts := breakupline(line);
  code := strtointdef('$'+parts.parts[0],$ff);
  t := unixtoole(unixtimefloat);
  decodedate(t,y,m,d);
  decodetime(t,hr,min,sec,ms);
  case code of
    $01 : begin
      writeln('nodehardware detected, synchroniasing time to '+d2d(y mod 100)+' '+d2d(m)+' '+d2d(d)+' '+d2d(hr)+' '+d2d(min)+' '+d2d(sec));


      lserial.sendstr('01 '+d2d(y mod 100)+' '+d2d(m)+' '+d2d(d)+' '+d2d(hr)+' '+d2d(min)+' '+d2d(sec)+' '+#13#10);
    end;
        $02 : begin
          //line informing that the RF board is starting a network, currently no special processing
        end;
        $10 : begin
          //line containing data from node, currently no special processing (will be processed later by processing log files)
        end;
  end;
  logfilename := d2d(y mod 100)+d2d(m)+d2d(d)+'_'+d2d(hr)+d2d(min);
  if fileexists(logfilename) then begin
    assignfile(f,logfilename);
    reset(f,1);
        seek(f,filesize(f));
  end else begin
    assignfile(f,logfilename);
    rewrite(f,1);
  end;
  logline := line + #13#10;
  blockwrite(f,logline[1],length(logline));
  closefile(f)
end;



procedure tsc.serialdataavailable(sender: tobject;error : word);
var
  termpos : integer;
  linelen : integer;
  line: string;
  incoming : string;
  i : integer;
```

```
begin
  incoming := lserial.receivestr;
  //lserial.sendstr(incoming);
  buf := buf + incoming;

  //writeln(makehexstring(incoming));
  while true do begin
    termpos := 0;
    for i := 1 to length(buf) do begin
      if (buf[i] = #10) or (buf[i] = #13) then begin
            termpos := i;
            break;
          end;
    end;
    if termpos = 0 then break;

        writeln('processing line');
        linelen := termpos -1;

    if linelen > 0 then begin
      line := copy(buf,1,linelen);
          processserialline(line);
          writeln(line);
        end;
        buf := copy(buf,termpos+1,maxlongint);


  end;

end;

type
  ttimehandler = class
    procedure timehandler(sender: tobject);
  end;

var
  routerpower : boolean;
  line : cint;

procedure ttimehandler.timehandler(sender: tobject);
var
  t : extended;
  y,m,d,hr,min,sec,ms : word;
begin
  t := unixtoole(unixtimefloat);
  decodedate(t,y,m,d);
  decodetime(t,hr,min,sec,ms);
  writeln(hr);
  routerpower := (hr >=9) and (hr < 16) ;

  line := TIOCM_RTS;
  //note: handshake lines on FTDI chip are inverted so a clear makes them go high and a set makes them go low.
  if routerpower then begin
    fpioctl(lserial.fdhandlein,TIOCMBIS,@line);
    writeln('router power on');
  end else begin
    fpioctl(lserial.fdhandlein,TIOCMBIC,@line);
    writeln('router power off');
  end;
end;

var
  timer: tltimer;
begin
  lcoreinit;
  lserial := tlserial.create(nil);
  lserial.baudrate := 19200;
  lserial.device := paramstr(1);
  lserial.ondataavailable := sc.serialdataavailable;
  lserial.open;

  writeln(lserial.fdhandlein);
  line := TIOCM_DTR;
  //note: handshake lines on FTDI chip are inverted so a clear makes them go high and a set makes them go low.
  fpioctl(lserial.fdhandlein,TIOCMBIC,@line);
  writeln('nodehardware placed in reset');
  sleep(1000);
  fpioctl(lserial.fdhandlein,TIOCMBIS,@line);
  writeln('nodehardware released from reset');

  timer := tltimer.create(nil);
  timer.ontimer := ttimehandler(nil).timehandler;
  timer.interval := 60000; //1 minute
  timer.enabled := true;

//  lserial.sendstr('test'#13#10);
  btime.init;
```

```
  messageloop;

end.
```

Appendix M

Source code for processing of data from nodes

This appendix contains the source code used to process the data collected by the nodes described in chapter 7. The code was written from scratch by the author in Java. The source code is presented below and can also be found in the file deploymentprocess.java in the "deploymentprocess" directory on the disc included for the examiners.

```
import java.io.*;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.*;

public class Deploymentprocess {

    private static final int numberoffields = 7;

        private static int[] messagelossdenominators5dayblock;


        private static double[] rssitable = new double[256];

        //the microchip datasheer provides a table converting power values to RSSI values. We need the opposite so we interpolate microchip's
        //figures
        private static void buildrssitable(int minpower, int ...rssivalues) {
                int numpoints = rssivalues.length;

                if (rssivalues[0] != 0) throw new RuntimeException("First rssi value must be zero");
                int prevrssi = 0;
                double prevpower = minpower;
                rssitable[0]=minpower;

                if (rssivalues[numpoints-1] != 255) throw new RuntimeException("Last rssi value must be 255");

                for (int i=1;i<numpoints;i++) {
                        int power = minpower+i;
                        int rssi=rssivalues[i];
                        rssitable[rssi]=power;

                        if ((rssi-prevrssi) > 1) {

                                for (int j=prevrssi+1;j<rssi;j++) {
                                        double weight = ((double)(j-prevrssi)) / (rssi-prevrssi);
                                        rssitable[j] = weight*power + (1-weight)*prevpower;
                                }
                        }
                        prevrssi = rssi;
                        prevpower = power;
                }
        }

        public static void main(String[] args) {
```

```
        buildrssitable(-90,0,1,2,5,9,13,18,23,27,32,37,43,48,53,58,63,68,73,78,83,89,95,100,107,111,117,121,125,129,
          133,138,143,148,153,159,165,170,176,183,188,193,198,203,207,212,216,221,225,228,233,239,245,250,253,254,255);
        //for (int i=0;i<256;i++){
        //        System.out.println(i+" "+rssitable[i]);
        //}


        File dir= new File("Y:\\deploymentrecordings");
        String start = "120315_0959"; //first file to read
//String end = "120727_1300"; //last file to read
        String end = "120816_0310";
//String start = end;
final String outputfilename = start+'-'+end+".csv";
final String outputfilenameano = start+'-'+end+"-allnodesonly.csv";
final String outputfilenameni = start+'-'+end+"-nointerpolation.csv";

final String outputfilename5da = start+'-'+end+"-5dayaverages.csv";

String[] filesfromos = dir.list();
ArrayList<String> Filestoprocess = new ArrayList<String>();
for (int i=0;i<filesfromos.length;i++) {
        String filename = filesfromos[i];
        //System.out.println(filename);
        if ((filename.length()==11) && (filename.charAt(6) == '_') && (filename.compareTo(start) >= 0) && (filename.compareTo(end) <= 0)) {
                //System.out.println(filename+"!!!!!");
                Filestoprocess.add(filename);
        }
}
Collections.sort(Filestoprocess);

ArrayList<String> lines = new ArrayList<String>();
ArrayList<String> nodesseen = new ArrayList<String>();

DateFormat dateformat = new SimpleDateFormat("yyMMdd_HHmm");
dateformat.setTimeZone(TimeZone.getTimeZone("Etc/UTC"));
Date experimentstart;
try {
                experimentstart = dateformat.parse(start);
        } catch (ParseException e2) {
                e2.printStackTrace();
                return;
        }
rounddatetime10m(experimentstart);

//first pass through the data, read the files and figure out what nodes exist
for (int i=0;i<Filestoprocess.size();i++) {

        String filename = Filestoprocess.get(i);
        //System.out.println(filename);


        //System.out.println(datetime);
        try {
                BufferedReader  r = new BufferedReader(new FileReader(new File(dir,filename)));

                String line;

                        while ((line = r.readLine()) != null) {
                                //System.out.println(line);
                                lines.add(filename+" "+line);
                                String[] linetokens = line.split(" ");
                                if (linetokens[0].equals("10") ) {
                                        //System.out.println("node data line");
                                        String longaddress = linetokens[1];
                                        if (!nodesseen.contains(longaddress)) nodesseen.add(longaddress);
                                }
                        }

                r.close();
        } catch (FileNotFoundException e) {
                        e.printStackTrace();
                        return;
                } catch (IOException e) {
                        e.printStackTrace();
                        return;
                }
}
Collections.sort(nodesseen);
System.out.println("list of nodes seen");
for (int i=0; i<nodesseen.size();i++) {
        System.out.println(nodesseen.get(i));
        }

long[] timesincelastseen = new long[nodesseen.size()];
for (int i=0; i<nodesseen.size();i++) {
        timesincelastseen[i] = -1;
        }

Date epoch = new Date(0);
```

```
Date currentdatetime = epoch;

final int startfields = 6;



        double [] temp1 = new double[nodesseen.size()];
        double [] temp2 = new double[nodesseen.size()];
        double [] soilres = new double[nodesseen.size()];
        double [] waterpresense = new double[nodesseen.size()];
        double [] lqi = new double[nodesseen.size()];
        double [] rssi = new double[nodesseen.size()];

        ArrayList<double []> outputdata = new ArrayList<double []>();

//second pass, gather data on each node at each timestep and produce output table
for (int i=0; i<lines.size();i++) {
        String line = lines.get(i);
        String[] linetokens = line.split(" ");

            if (linetokens[1].equals("10") ) {


            Date datetime;
            try {
                    datetime = dateformat.parse(linetokens[0]);
            } catch (ParseException e1) {
                    e1.printStackTrace();
                    return;
            }

                rounddatetime10m(datetime);

                if (!currentdatetime.equals(datetime)) {
                    if (!currentdatetime.equals(epoch)) {
                            fillandaddline(outputdata,experimentstart,nodesseen,currentdatetime,
                                    timesincelastseen,temp1,temp2,soilres,waterpresense, lqi, rssi, startfields);
                    }
                    long timedifference = (datetime.getTime() - currentdatetime.getTime())/60000;
                    for (int j=0; j<nodesseen.size();j++) {
                    if (timesincelastseen[j] != -1) timesincelastseen[j] += timedifference;
                    }
                    currentdatetime = datetime;
                }


                    //System.out.println("node data line");
                    String longaddress = linetokens[2];
                    int nodeno = nodesseen.indexOf(longaddress);

                    timesincelastseen[nodeno] = 0;

                    int temp1raw = Integer.parseInt(linetokens[3],16);
                    int temp2raw = Integer.parseInt(linetokens[4],16);
                    int soilresraw = Integer.parseInt(linetokens[5],16);
                    int waterpresenseraw = Integer.parseInt(linetokens[6],16);

                    int lqiraw = Integer.parseInt(linetokens[8],16);
                    int rssiraw = Integer.parseInt(linetokens[9],16);

                    double temp1millivolts = (temp1raw * 3300.0) / 1024;
                    double temp2millivolts = (temp2raw * 3300.0) / 1024;

                    temp1[nodeno] = (temp1millivolts - 400) / 19.5;
                temp2[nodeno] = (temp2millivolts - 400) / 19.5;

                int soilresshort = 0x1AC;
                int soilres10k = 0xBc;

                int soilresoffset = soilresshort - (soilres10k * 2);

                //cancel offset from diode;
                double soilresoc = soilresraw+soilresoffset;
                double soilresshortoc = soilresshort+soilresoffset;

                double soilresdividerfraction = soilresoc / soilresshortoc;
                //System.out.println(soilresdividerfraction);

                soilres[nodeno] = (10000/soilresdividerfraction) - 10000;

                waterpresense[nodeno] = waterpresenseraw;

                lqi[nodeno] = lqiraw;
                rssi[nodeno] = rssitable[rssiraw];

            }

}
```

```
        //add final line to outputdata
fillandaddline(outputdata,experimentstart,nodesseen,currentdatetime, timesincelastseen,temp1,temp2,soilres,waterpresense, lqi, rssi, startfields);

int nodecount = nodesseen.size();

//perform averaging for missing data points
for (int i=0;i<outputdata.size();i++) {
        double[] line = outputdata.get(i);
        for (int j=0; j<nodecount;j++) {
                if (line[startfields+j] != 0) {
                        int lowi = i;
                        while ((lowi >= 0) && (outputdata.get(lowi)[startfields+j] != 0)) {
                                lowi --;
                        }
                        int highi = i;
                        while ((highi < outputdata.size()  && (outputdata.get(highi)[startfields+j] != 0))) {
                                highi ++;
                        }

                        double weight; //weight of average, 0 means use entirely value from lowi
                                       //                   1 means use entierly value from highi


                        if (lowi < 0) {
                                //we haven't seen the node yet, use the value from the first time we see it
                                lowi = 0;
                                weight = 1;
                        } else if (highi >= outputdata.size()) {
                                //we don't see the node again, use the value from the last time we saw it
                                highi = outputdata.size() -1;
                                weight = 0;
                        } else {
                                double lowtime = outputdata.get(lowi)[startfields-1];
                                double hightime = outputdata.get(highi)[startfields-1];
                                double time = line[startfields-1];
                                weight = (time-lowtime) / (hightime-lowtime);
                        }
                        double[] lowline = outputdata.get(lowi);
                                double[] highline = outputdata.get(highi);
                        for (int k=1;k<numberoffields;k++) {

                                        line[startfields+(k*nodecount)+j] =
                                          weight *highline[startfields+(k*nodecount)+j]) +
                                          ((1-weight) *lowline[startfields+(k*nodecount)+j]);
                        }

                }
        }
}


try {
                writeoutputfile(dir, outputfilename, nodesseen, outputdata, false, nodecount, startfields, true);
                writeoutputfile(dir, outputfilenameano, nodesseen, outputdata, true, nodecount, startfields, true);
                writeoutputfile(dir, outputfilenameni, nodesseen, outputdata, true, nodecount, startfields, false);
} catch (FileNotFoundException e) {
                e.printStackTrace();
                return;
        }

int max5dayblock = (int)(outputdata.get(outputdata.size()-1)[5]/5);
messagelossdenominators5dayblock = new int [max5dayblock+1];
double messageloss5dayblock[][] = new double [max5dayblock+1][nodecount];
for (int i=0;i<=max5dayblock;i++) {
    Arrays.fill(messageloss5dayblock[i], -1);
}
int messagelossnumeratorsalltime[] = new int[nodecount];
int messagelossdenominatoralltime = 0;

int messagelossnumeratorscurrentblock[] = new int[nodecount];
int messagelossdenominatorcurrentblock = 0;

int current5dayblock = 0;
//Calculate message loss percentages on 5-day basis and for the complete lifetime of each node
for (int i=0;i<outputdata.size();i++) {
        double[] line = outputdata.get(i);
        if (line[4] != 0) continue; //ignore time points other than the expected hourly ones
        messagelossdenominatoralltime++;
        for (int j=0; j<nodecount;j++) {
                if (line[startfields+j] != 0) {
                        messagelossnumeratorsalltime[j]++;
                }
        }



        int new5dayblock = (int)(line[5]/5);
        if (new5dayblock != current5dayblock) {
```

```
                finish5dayblock(nodecount, messageloss5dayblock, messagelossnumeratorscurrentblock,
                                        messagelossdenominatorcurrentblock,
                                        current5dayblock);
                messagelossdenominatorcurrentblock =0;
                Arrays.fill(messagelossnumeratorscurrentblock, 0);
                current5dayblock = new5dayblock;
        }

        messagelossdenominatorcurrentblock++;
        for (int j=0; j<nodecount;j++) {
                if (line[startfields+j] != 0) {
                        messagelossnumeratorscurrentblock[j]++;
                }
        }
}
finish5dayblock(nodecount, messageloss5dayblock, messagelossnumeratorscurrentblock,
                        messagelossdenominatorcurrentblock,
                        current5dayblock);
//System.out.println(messagelossnumeratorsalltime[0]);
//System.out.println(messagelossdenominatoralltime);
for (int j=0; j<nodecount;j++) {
            System.out.println(((double)messagelossnumeratorsalltime[j])/messagelossdenominatoralltime*100);

    }

PrintStream out;
try {
                out = new PrintStream(new File (dir,outputfilename5da));
        } catch (FileNotFoundException e) {
                e.printStackTrace();
                return;
        }
        for (int j=0; j<nodesseen.size();j++) {
                out.print(","+nodesseen.get(j));
        }
        out.print(",\"time points used\"");
        out.println();
        for (int i=0;i<=max5dayblock;i++) {
                out.print((i*5)+2.5);
                if (messageloss5dayblock[i][0] != -1) {

                        for (int j=0; j<nodecount;j++) {
                                out.print(","+messageloss5dayblock[i][j]*100);
                        }
                        out.print(","+messagelossdenominators5dayblock[i]);
                } else {
                        for (int j=0; j<nodecount;j++) {
                                out.print(",");
                        }
                        out.print(",0");
                }
                out.println();
        }

        out.close();

}



private static PrintStream writeoutputfile(File dir, String outputfilename,
                ArrayList<String> nodesseen, ArrayList<double[]> outputdata,
                boolean allnodesonly, int nodecount, int startfields, boolean includeinterpolated)
throws FileNotFoundException {
        PrintStream out;
        out = new PrintStream(new File (dir,outputfilename));


        //print first header line
        out.print(",,,,,");

        out.print(",time since last seen");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
        }

        out.print(",temperature sensor 1");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
        }

        out.print(",temperature sensor 2");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
        }

        out.print(",soil resistance");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
```

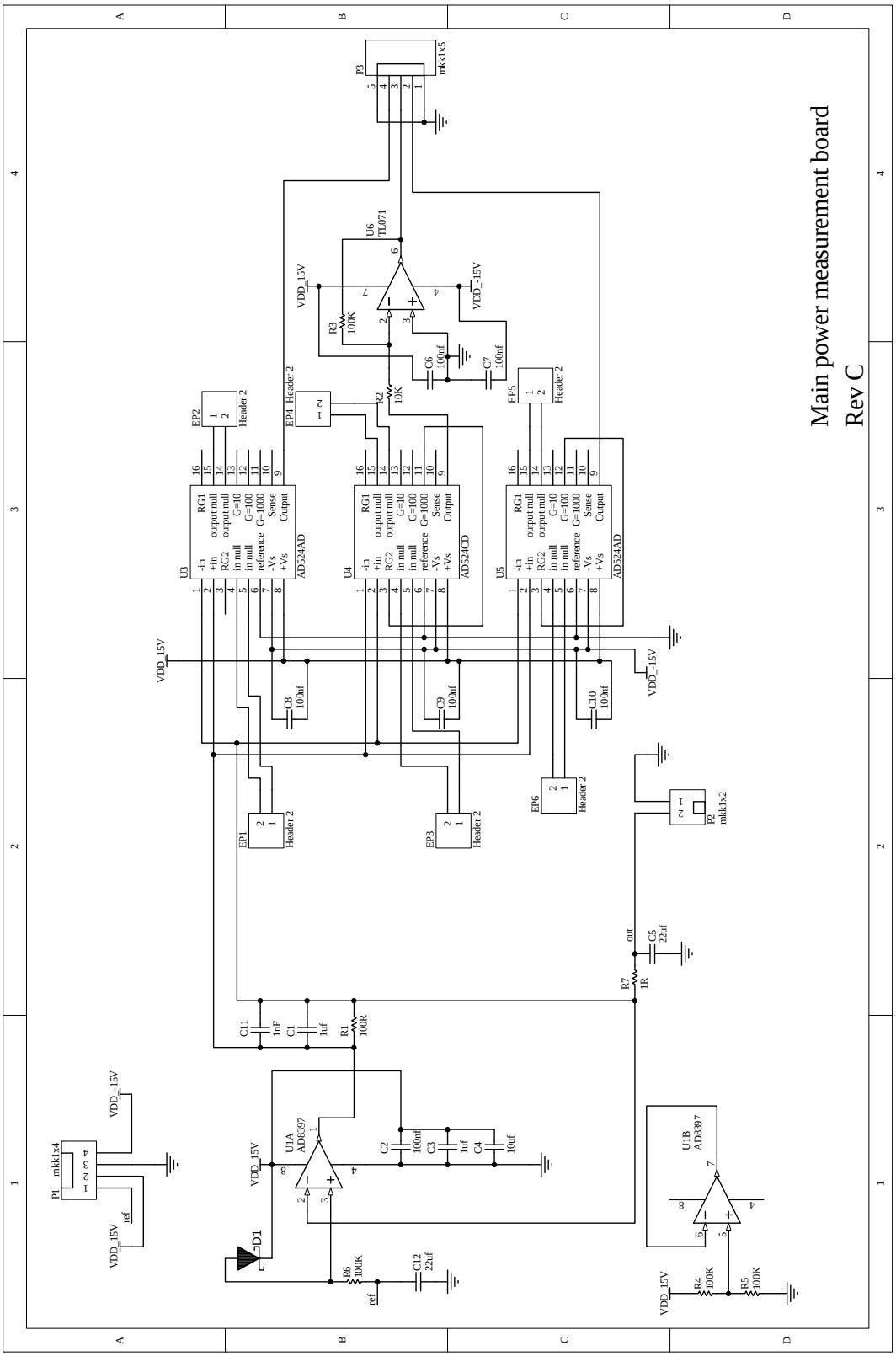
```
        }

        out.print(",water presense");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
        }

        out.print(",lqi");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
        }

        out.print(",rssi");
        for (int j=1; j<nodesseen.size();j++) {
                out.print(",");
        }


        out.println();

        //print second header line
        out.print("year,month,day,hour,minuite,daycount");

        for (int i=0; i<numberoffields;i++) {
                for (int j=0; j<nodesseen.size();j++) {
                        out.print(","+nodesseen.get(j));
                }
        }
        out.println();

        //actually write out data
        for (int i=0;i<outputdata.size();i++) {
                Boolean outputThisLine = true;
                double[] line = outputdata.get(i);
                if (allnodesonly) {

                        for (int j=0; j<nodecount;j++) {
                        if (line[startfields+j] != 0) {
                                    //HACK, ignore some nodes since if we don't we get no lines in the result.
                                String nodelongaddress =nodesseen.get(j);
                                //Intern the string so we can compare it using != operator
                                //nodelongaddress = nodelongaddress.intern();
                                int nodeno = Integer.parseInt(nodelongaddress.substring(8));
                                if  ((nodeno > 6 ) && (nodeno < 13)){

                                        outputThisLine = false;
                                }
                        }
                        }
                }
                if (outputThisLine) {

                        out.print(line[0]);
                        for (int j=1;j<line.length;j++) {
                                boolean outputthisvalue=true;
                                if (!includeinterpolated && (j >= startfields + nodecount)) {
                                        int nodeindex = (j - startfields) % nodecount;
                                        double timesincelastseen = line[startfields+nodeindex];
                                        if (timesincelastseen != 0) outputthisvalue = false;
                                }
                                out.print(",");
                                if (outputthisvalue) out.print(line[j]);
                        }
                        out.println();
                }
        }
        out.close();
        return out;
}

private static void finish5dayblock(int nodecount, double[][] messageloss5dayblock,
                int[] messagelossnumeratorscurrentblock,
                int messagelossdenominatorcurrentblock,
                int current5dayblock) {
        for (int j=0; j<nodecount;j++) {
                messageloss5dayblock[current5dayblock][j] = ((double)messagelossnumeratorscurrentblock[j])/messagelossdenominatorcurrentblock;
                messagelossdenominators5dayblock[current5dayblock] = messagelossdenominatorcurrentblock;
        }
}

//round date/time stored in a Date object to the nearest 10 minutes.
private static void rounddatetime10m(Date datetime) {
        //datetime.setTime(Math.round(((double)datetime.getTime())/600000 )*600000  );
        //widen this to 20 minuites due to time drift issues
        datetime.setTime(Math.round(((double)datetime.getTime())/1800000 )*1800000  );
}

static Calendar c = new GregorianCalendar(TimeZone.getTimeZone("Etc/UTC"));
```

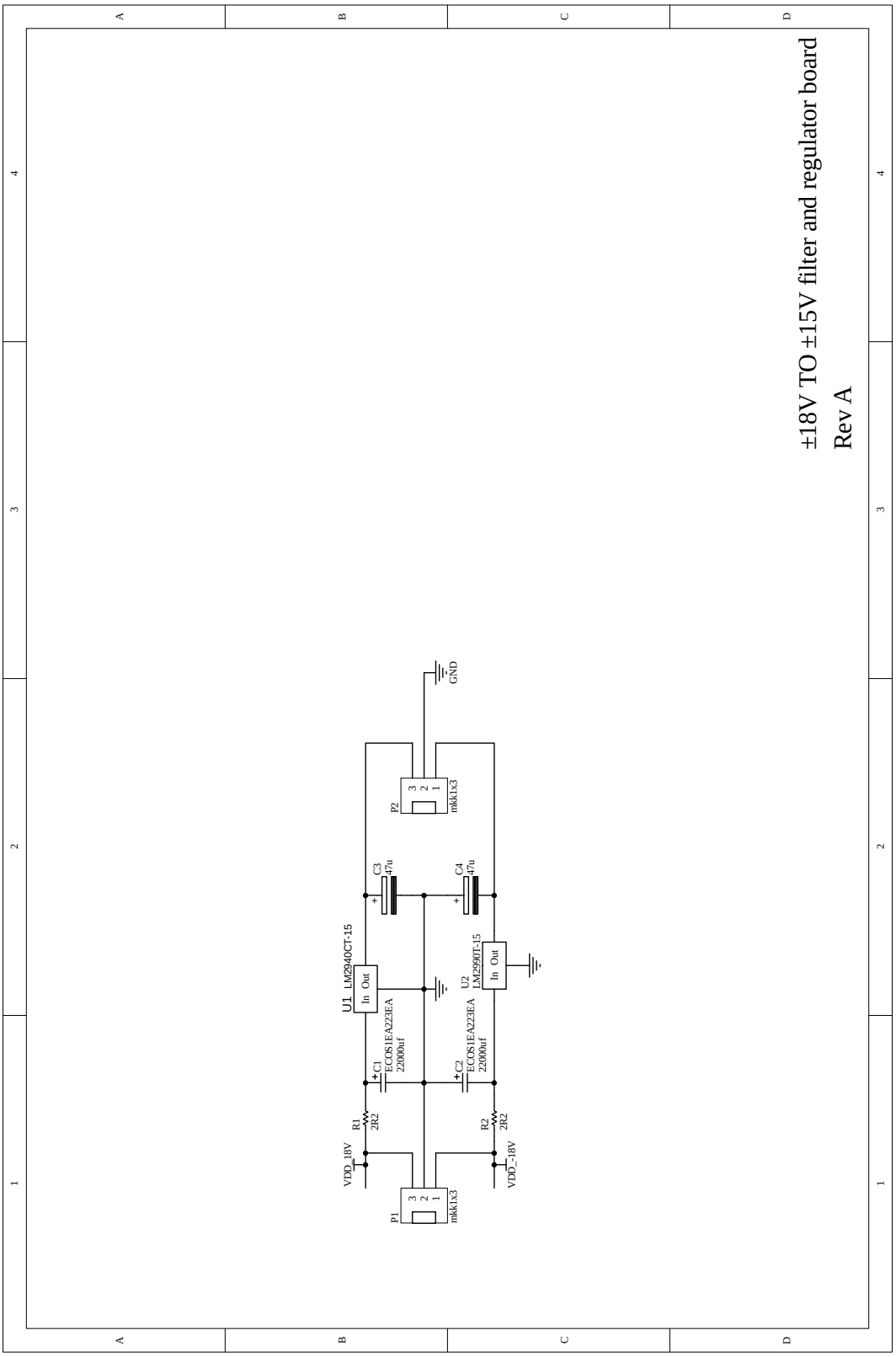
```
	private static void fillandaddline(ArrayList<double[]> outputdata, Date experimentstart,
	  ArrayList<String> nodesseen, Date currentdatetime, long[] timesincelastseen, double[] temp1,
	  double[] temp2, double[] soilres, double[] waterpresense, double[] lqi, double[] rssi, int startfields) {
		long timesinceexperimentstart = (currentdatetime.getTime() - experimentstart.getTime())/60000;

		int nodecount = nodesseen.size();
		double [] line = new double [(nodesseen.size()*numberoffields)+startfields];

		c.setTime(currentdatetime);
		line[0] = c.get(Calendar.YEAR);
		line[1] = c.get(Calendar.MONTH)+1; //for some reason the java calendar class numbers months starting at 0
		line[2] = c.get(Calendar.DAY_OF_MONTH);
		line[3] = c.get(Calendar.HOUR_OF_DAY);
		line[4] = c.get(Calendar.MINUTE);
		line[5] = ((double)timesinceexperimentstart) / 1440;
		for (int j=0; j<nodecount;j++) {
			line[startfields+j] = timesincelastseen[j];
			line[startfields+nodecount+j] = temp1[j];
			line[startfields+(nodecount*2)+j] = temp2[j];
			line[startfields+(nodecount*3)+j] = soilres[j];
			line[startfields+(nodecount*4)+j] = waterpresense[j];
			line[startfields+(nodecount*5)+j] = lqi[j];
			line[startfields+(nodecount*6)+j] = rssi[j];
		}
		outputdata.add(line);
	}

}
```

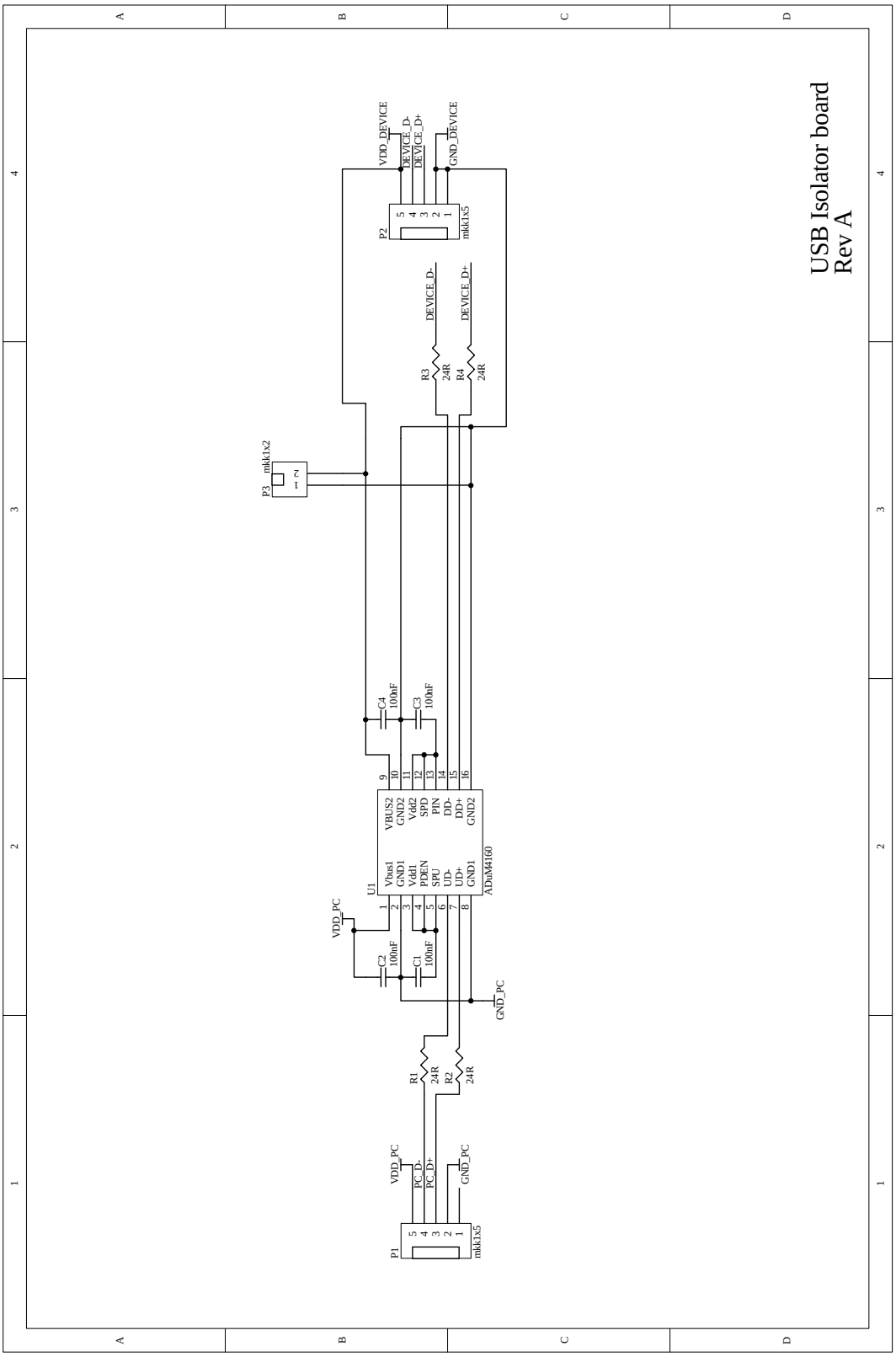
Appendix N

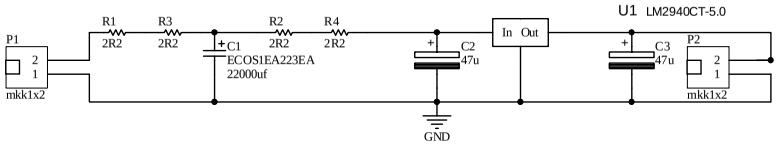
Power measurement schematics

This appendix contains the schematics for the main power measurement board, the power filtering boards and the USB isolator board. The full design files for the boards are included in the directories “wsnpowermeasurement-revb”, “18-15reg”, “usbisolator” and “18-5reg” on the disc included for the examiners.


Main power measurement board
Rev C
P3
mkk1x5
U6
TL071
VDD_15V
VDD_-15V
R3
100K
R2
10K
C6
100nf
C7
100nf
EP4 Header 2
EP2
Header 2
EP5
Header 2
U3
U4
U5
AD524AD
AD524CD
C8
100nf
C9
100nf
C10
100nf
EP1
Header 2
EP3
Header 2
EP6
Header 2
P2
mkk1x2
out
C5
22uf
R7
1R
C11
1nF
C1
1uf
R1
100R
P1
mkk1x4
ref
U1A
AD8397
U1B
AD8397
C2
100nf
C3
1uf
C4
10uf
D1
R6
100K
C12
22uf
R4
100K
R5
100K


±18V TO ±15V filter and regulator board
Rev A
A
B
C
D
1
2
3
4
GND
P2
mkk1x3
C3
47u
C4
47u
U1 LM2940CT-15
In Out
U2
LM2990T-15
In Out
C1
ECOS1EA223EA
22000uf
C2
ECOS1EA223EA
22000uf
R1
2R2
R2
2R2
VDD_18V
VDD_-18V
P1
mkk1x3


USB Isolator board
Rev A
VDD_DEVICE
DEVICE_D-
DEVICE_D+
GND_DEVICE
P2
mkk1x5
DEVICE_D-
DEVICE_D+
R3
24R
R4
24R
P3 mkk1x2
C4
100nF
C3
100nF
U1
ADuM4160
VBUS2
GND2
Vdd2
SPD
PIN
DD-
DD+
GND2
Vbus1
GND1
Vdd1
PDEN
SPU
UD-
UD+
GND1
VDD_PC
C2
100nF
C1
100nF
GND_PC
R1
24R
R2
24R
VDD_PC
PC_D-
PC_D+
GND_PC
P1
mkk1x5



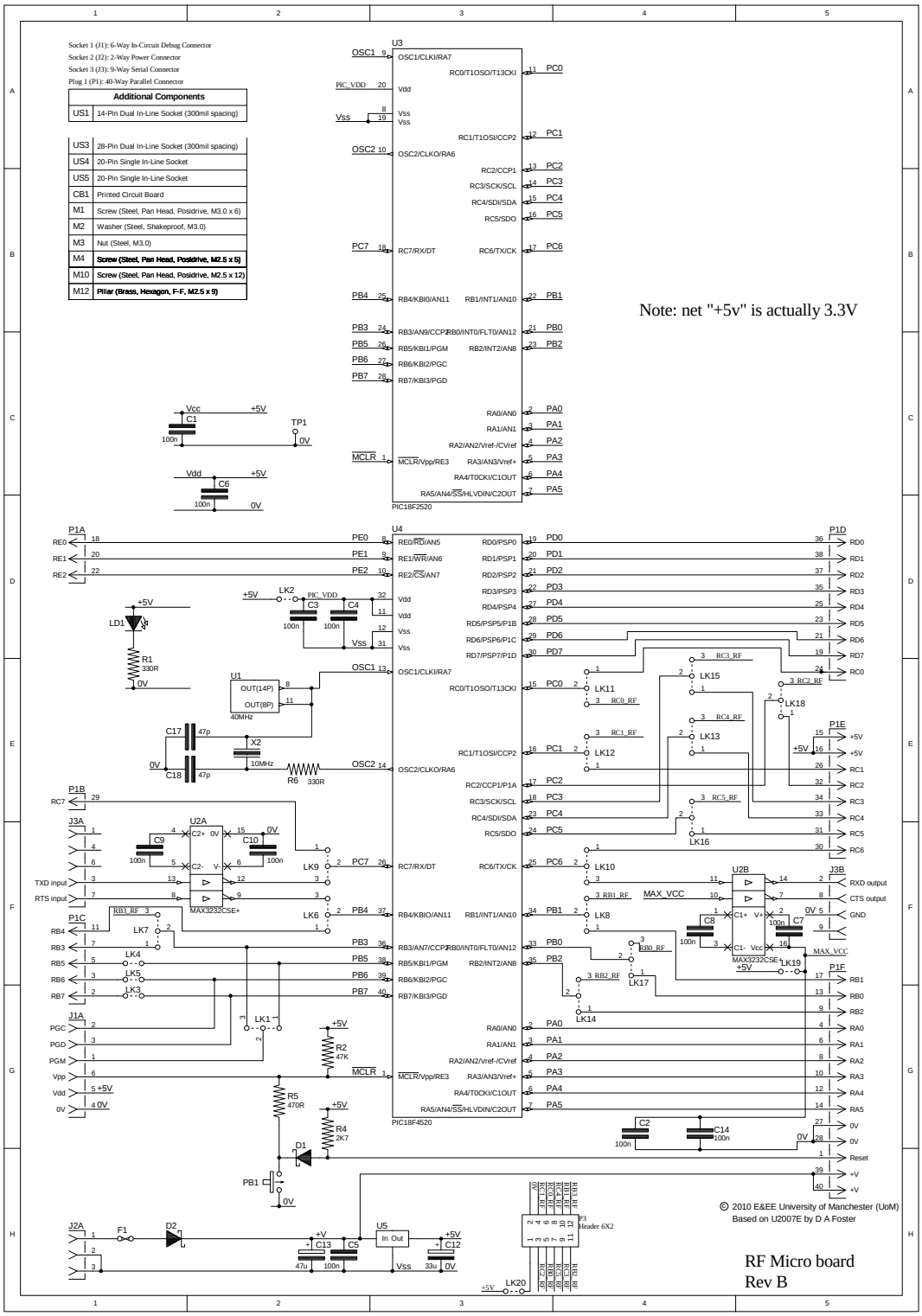
18V-5V filter and regulator board for powering isolated USB

Rev A

Appendix O

RF micro board schematic

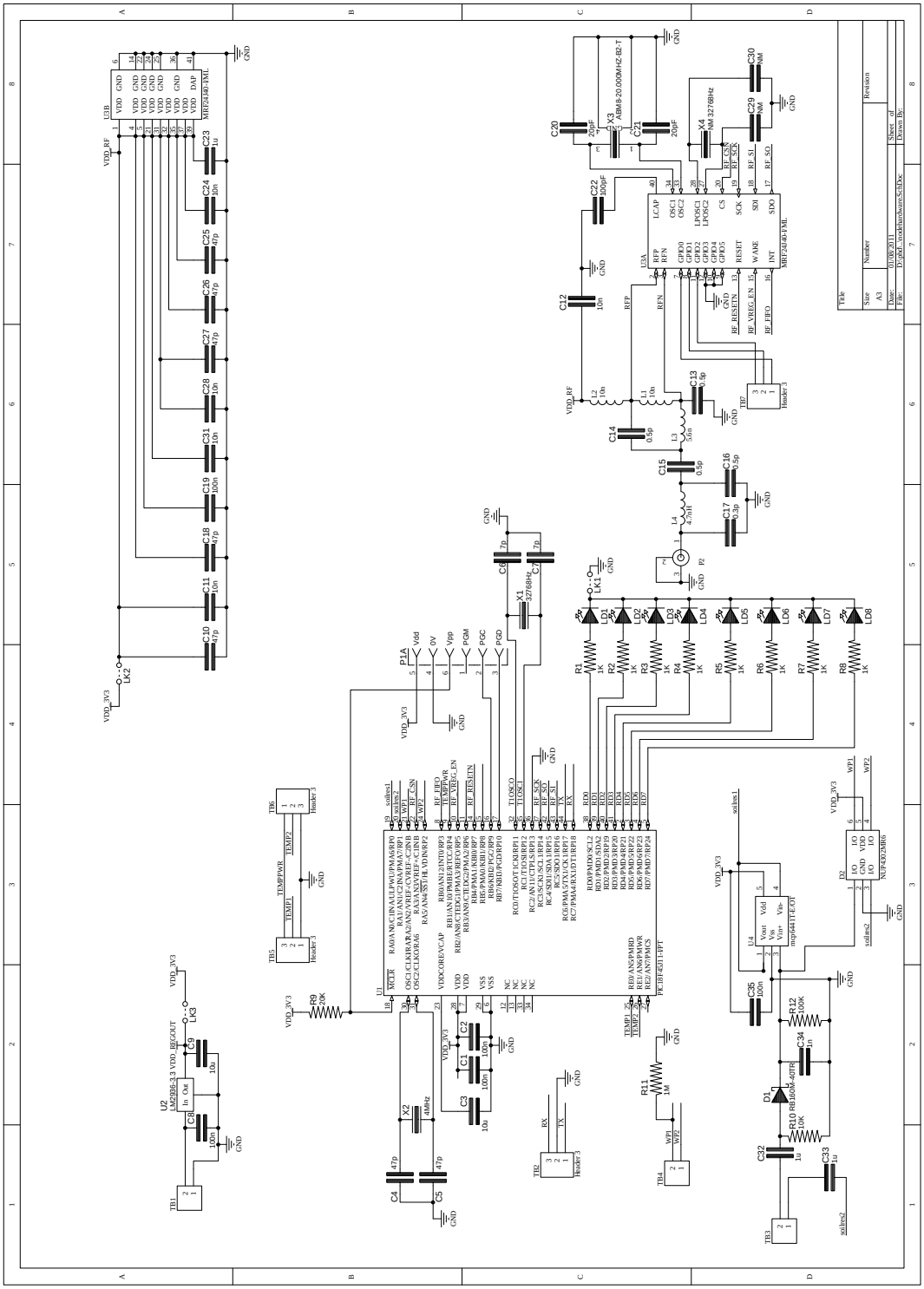
This appendix contains the schematic for revision B of the RF micro board. The full design files are available on the disc provided to the examiners in the directory "rfmb-revb".


Socket 1 (J1): 6-Way In-Circuit Debug Connector
Socket 2 (J2): 2-Way Power Connector
Socket 3 (J3): 9-Way Serial Connector
Plug 1 (P1): 40-Way Parallel Connector
Additional Components
US1 14-Pin Dual In-Line Socket (300mil spacing)
US3 28-Pin Dual In-Line Socket (300mil spacing)
US4 20-Pin Single In-Line Socket
US5 20-Pin Single In-Line Socket
CB1 Printed Circuit Board
M1 Screw (Steel, Pan Head, Posidrive, M3.0 x 6)
M2 Washer (Steel, Shakeproof, M3.0)
M3 Nut (Steel, M3.0)
M4 Screw (Steel, Pan Head, Posidrive, M2.5 x 5)
M10 Screw (Steel, Pan Head, Posidrive, M2.5 x 12)
M12 Pillar (Brass, Hexagon, F-F, M2.5 x 9)
U3
PIC18F2520
U4
PIC18F4520
Note: net "+5v" is actually 3.3V
© 2010 E&EE University of Manchester (UoM)
Based on U2007E by D A Foster
RF Micro board
Rev B

Appendix P

Node hardware schematics

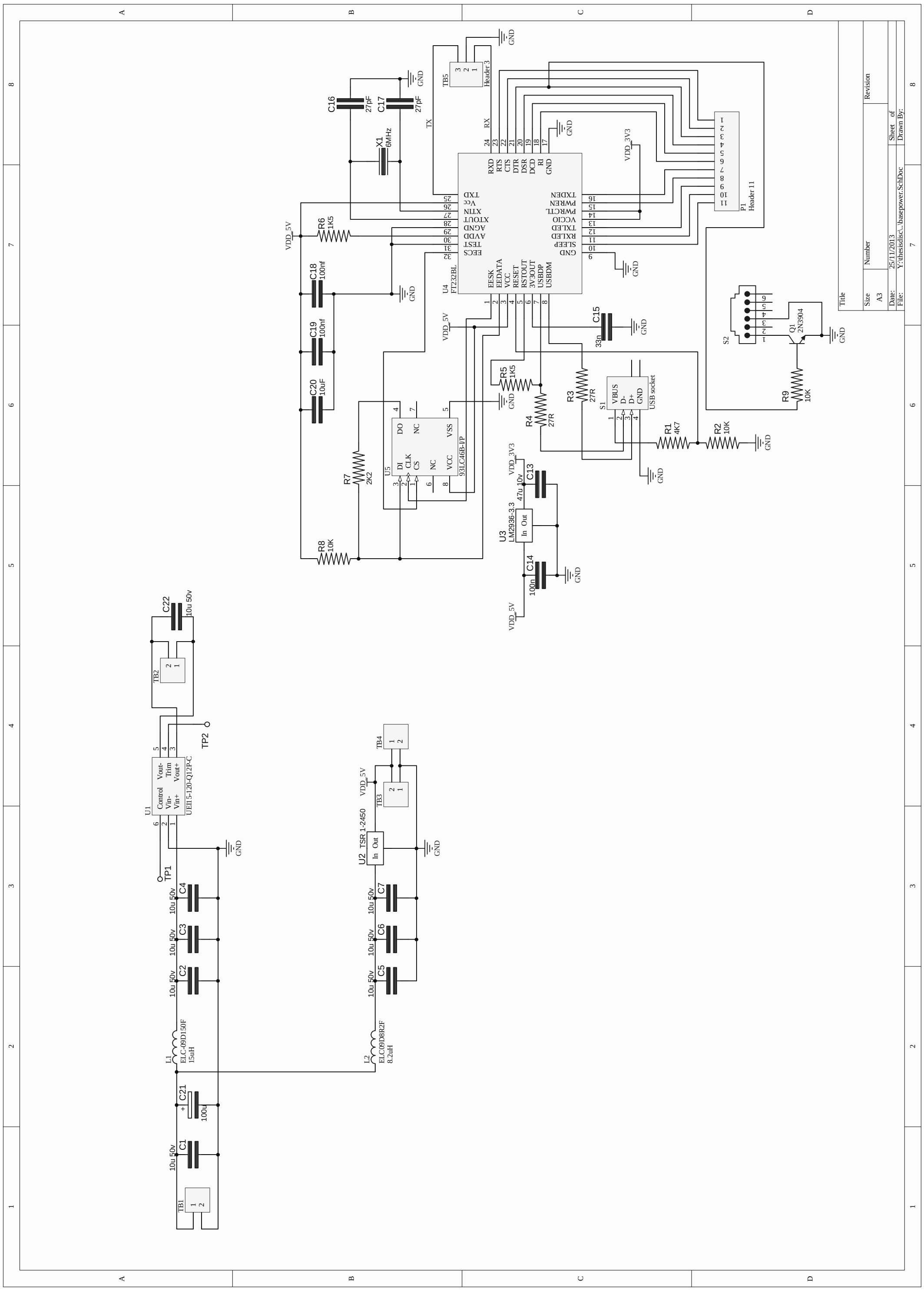
This appendix contains the schematic of the main node hardware board. The full design files are available on the disc provided to the examiners in the directory "nodehardware-revb".



Appendix Q

Base station power board schematics

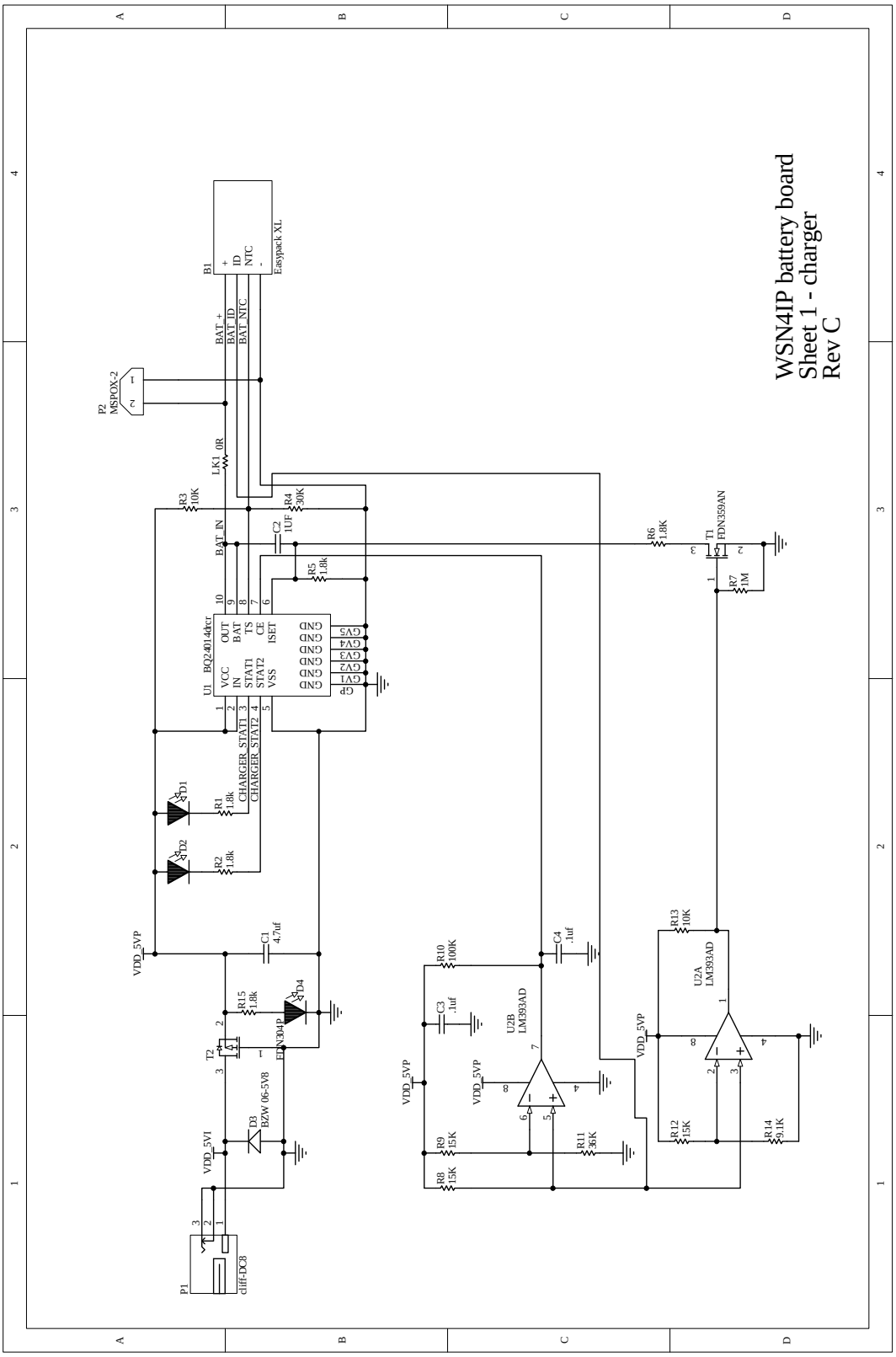
This appendix contains the schematic of the power and USB serial board used in the base station. The full design files are available on the disc provided to the examiners in the directory "basepower-revb".

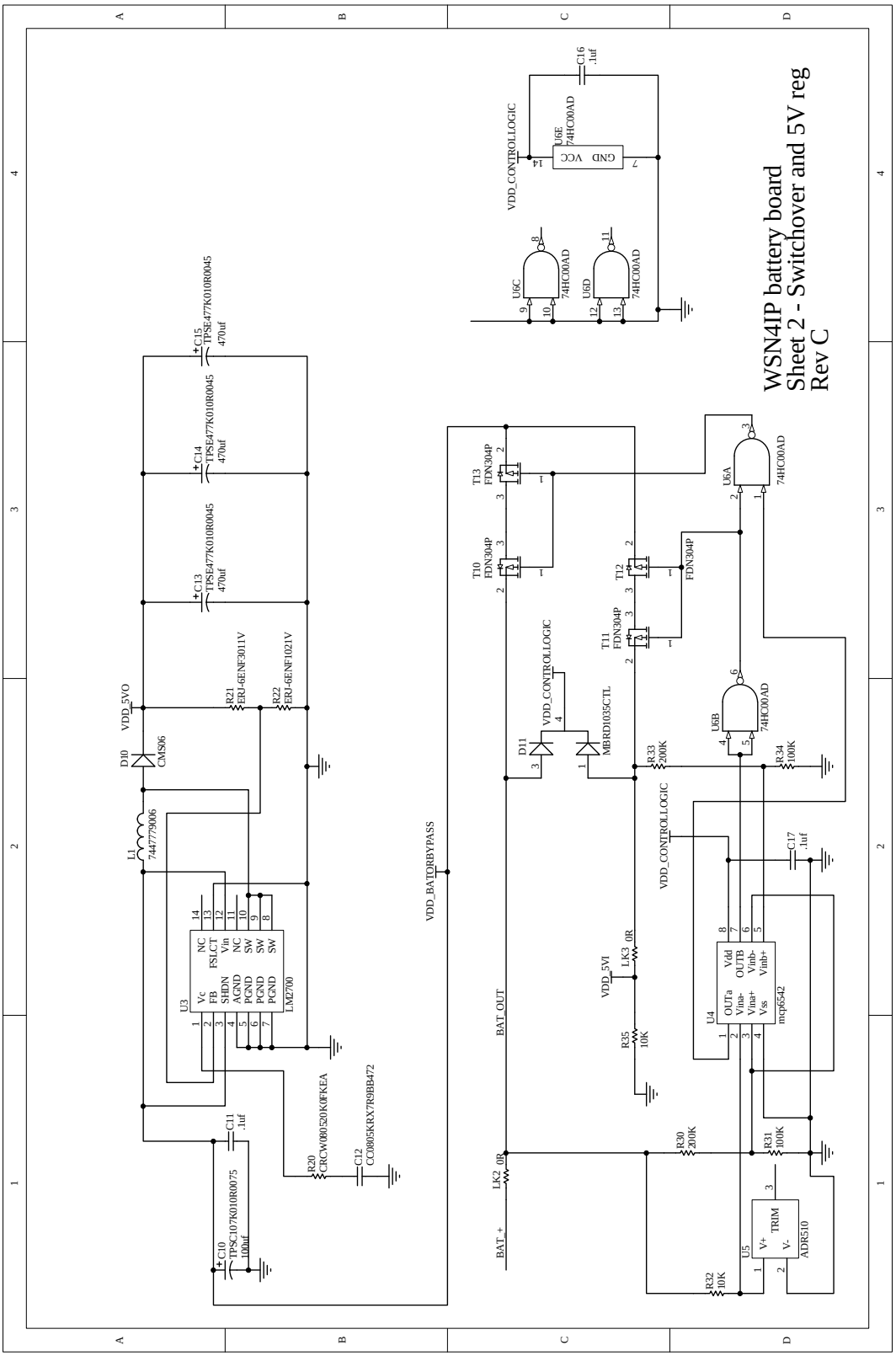

FT232BL
93LC46B-I/P
LM2936-3.3
TSR 1-2450
UEI15-120-Q12P-C
ELC-09D150F
ELC09D8R2F
2N3904
Header 11
Header3
USB socket
VDD_5V
VDD_3V3
GND
6MHz
Title
Size
A3
Number
Revision
Date:
25/11/2013
File:
Sheet of
Drawn By:

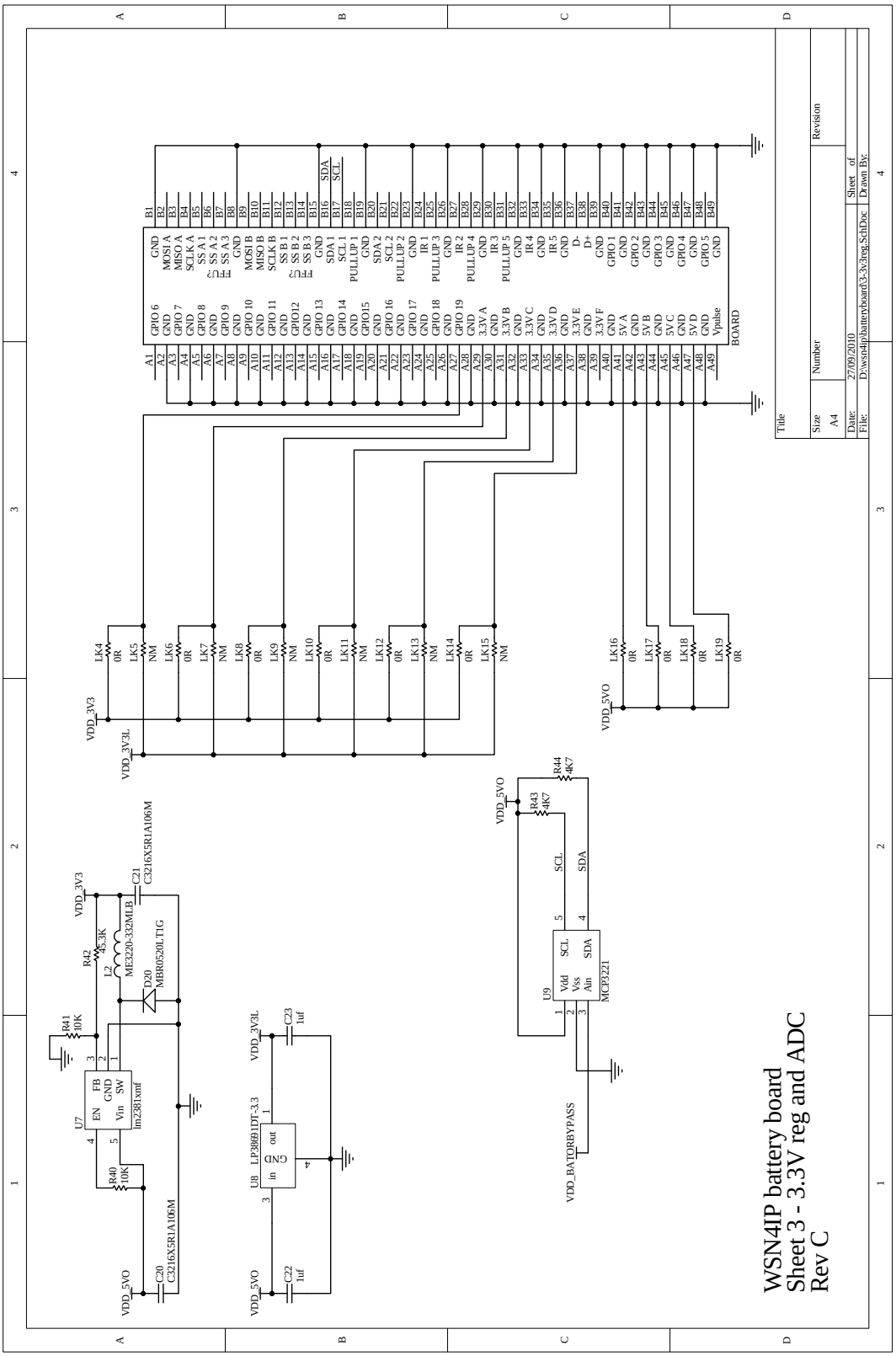
Appendix R

WSN4IP Battery board schematics

This appendix contains the schematics for revision C of the WSN4IP battery board. The full design files are available on the disc provided to the examiners in the directory "batteryboard-revc".


WSN4IP battery board
Sheet 1 - charger
Rev C
B1
Easypack XL
BAT +
BAT_ID
BAT_NTC
P2
MSPOX-2
LK1 0R
R3 10K
R4 300K
C2 1UF
BAT_IN
R5 1.8k
R6 1.8K
T1 FDN359AN
R7 1M
U1 BQ2401xdcr
VCC
IN
STAT1
STAT2
VSS
OUT
BAT
TS
CE
ISET
GND
CHARGER_STAT1
CHARGER_STAT2
D1
D2
R1 1.8k
R2 1.8k
VDD_5VP
C1 4.7uf
R15 1.8k
D4
T2 FDN304P
D3 BZW 06-5V8
VDD_5V1
P1
diff-DC8
R10 100K
C3 .1uf
U2B LM393AD
C4 .1uf
R9 15K
R8 15K
R11 36K
R13 10K
U2A LM393AD
R12 15K
R14 9.1K

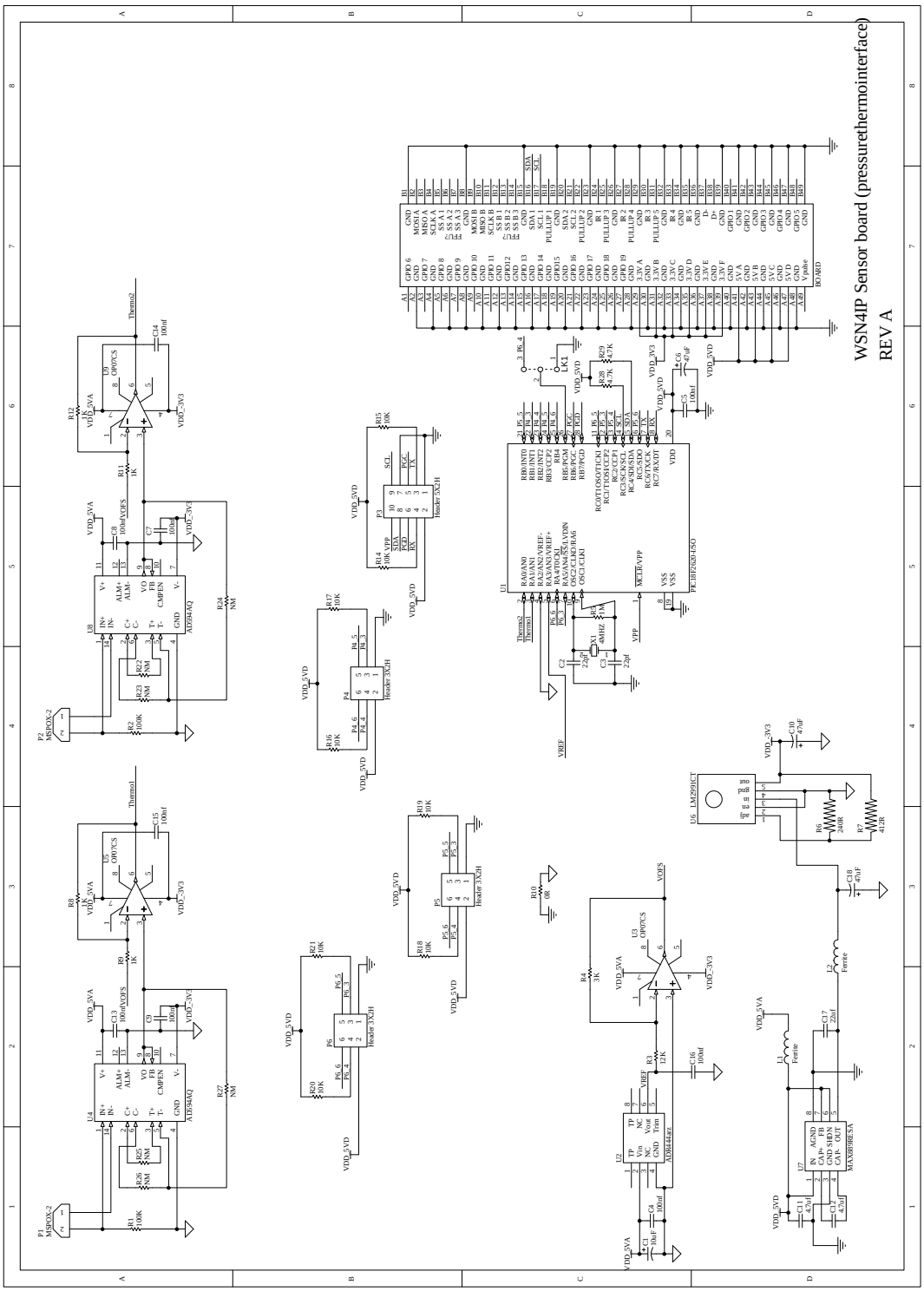

WSN4IP battery board
Sheet 2 - Switchover and 5V reg
Rev C
VDD_BATORBYPASS
VDD_5VO
VDD_CONTROLLOGIC
BAT_OUT
BAT_+
VDD_5VI
L1
7447779006
D10
CMSH6
U3
LM2700
C10
TPSC107K010R0075
100uf
C11
.1uf
C12
CC0805KRX7R9BB472
R20
CRCW0805205K0FKEA
R21
ERJ-6ENF3011V
R22
ERJ-6ENF1021V
C13
TPSE477K010R0045
470uf
C14
TPSE477K010R0045
470uf
C15
TPSE477K010R0045
470uf
D11
MBRD1035CTL
T10
FDN304P
T11
FDN304P
T12
FDN304P
T13
FDN304P
U6A
74HC00AD
U6B
74HC00AD
U6C
74HC00AD
U6D
74HC00AD
U6E
74HC00AD
C16
.1uf
C17
.1uf
R33
200K
R34
100K
R35
10K
R30
200K
R31
100K
R32
10K
LK2
0R
LK3
0R
U4
mcp6542
U5
ADR510


WSN4IP battery board
Sheet 3 - 3.3V reg and ADC
Rev C
BOARD
FFCU2
FFCU3
U7
lm2381xmf
U8
LP3869ILDT-3.3
U9
MCP3221
D20
MBR0520LT1G
L2
ME3220-332MLB
R40
10K
R41
10K
R42
45.3K
R43
4K7
R44
4K7
C20
C3216X5R1A106M
C21
C3216X5R1A106M
C22
1uf
C23
1uf
VDD_3V3
VDD_3V3L
VDD_5VO
VDD_BATORBYPASS
SCL
SDA
Title
Size
A4
Number
Revision
Date:
27/09/2010
Sheet of
File:
D:\wsn4ip\batteryboard\3-3v3reg.SchDoc
Drawn By:

Appendix S

WSN4IP Sensor board schematics

This appendix contains the schematics for revision A of the WSN4IP sensor board. The full design files are available on the disc provided to the examiners in the directory "pressurethermointerface-reva".


WSN4IP Sensor board (pressurethermointerface)
REV A