THE DEVELOPMENT OF A SPECTROMETER FOR PORTABLE NMR SYSTEMS

A thesis presented in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Physics at Massey University

Robin Dykstra

2006

Appendix D

MASSEY UNIVERSITY

Application for Approval of Request to Embargo a Thesis

(Pursuant to AC98/168 (Revised 2), Approved by Academic Board 17/02/99)

Name of Candidate: Robin Dykstra................................ID Number: 87021211......

Degree: PhD......................................Dept/Institute/School: IFS......................

Thesis title: The development of a spectrometer for portable NMR systems....................

....

Name of Chief Supervisor: Prof Paul Callaghan...........Telephone Ext:5333

As author of the above named thesis, I request that my thesis be embargoed from public access until (date) 1/7/2008 for the following reasons:

■ Thesis contains commercially sensitive information.

☐ Thesis contains information which is personal or private and/or which was given on the basis that it not be disclosed.

☐ Immediate disclosure of thesis contents would not allow the author a reasonable opportunity to publish all or part of the thesis.

☐ Other (specify):...

....

Please explain here why you think this request is justified:

Some of the material within the thesis is used as the basis for a series of new products which are being marketed through Magritek Ltd.

....

....

....

....

Signed (Candidate):....[illegible]..........................Date: 10/7/06

Endorsed (Chief Supervisor):....[illegible]..........................Date: 11/7/06

Approved/~~Not Approved~~ (Representative of VC):....[illegible]..........Date: 12/7/2006

Note: Copies of this form, once approved by the representative of the Vice-Chancellor, must be bound into <u>every</u> copy of the thesis.

Dedicated to the memory of my late father,
Michiel Dykstra,
who passed away during the preparation of this thesis.

Abstract

Nuclear Magnetic Resonance (NMR) is a relatively complex technique and normally requires expensive equipment. However, with advances in computing, electronics and permanent magnet technologies, NMR is becoming more feasible as a non-invasive tool for industry. The strength of NMR is its ability to probe at the molecular level and hence gain information about molecular structure, organisation, abundance and orientation. This thesis describes the development of an instrumentation platform technology that is compact and therefore portable. It has been produced to aid the development of NMR based tools or sensors for research and industry and will lead to a series of low cost, portable NMR systems for the non-destructive testing of materials such as polymer composites, rubber, timber, bricks and concrete.

The instrumentation is largely electronics based and consists of a series of modules that can be interconnected to produce a solution. The first of two main modules is called the system core. What is common to all NMR applications is the generation of precisely timed signals, the capturing of signals and the processing/display of data. This has been implemented by developing a general purpose Digital Signal Processor (DSP) based instrumentation and control module that uses a Universal Serial Bus interface to communicate with a host computer. A graphical user interface is provided by an application running under Windows® XP.

The second main module is a radio frequency transceiver that has been developed using digital receiver technology. The signals, after some amplification, are digitized with a 14-bit, 62.5 MHz analogue to digital converter. The sampled signal is then mixed digitally with synthesized sine and cosine functions to generate lower frequency quadrature outputs which are then digitally filtered and decimated before being passed onto the DSP for further processing and storage. A direct digital synthesizer with an analogue output is used to generate any required excitation signals. All synthesizers have phase and frequency hopping capabilities and are phase locked to each other and the DSP.

The system was designed to interface to a range of NMR probes. The type of probe is determined by the intended application and each probe has specific requirements such as the type of radio frequency power amplifier, duplexer and preamplifier needed. This results in a number of instrumentation variations and a modular instrument enclosure was used to cater for these variations. The instrument was first configured for an NMR probe called the NMR-MOUSE. Tests were performed with this probe to verify the correct operation and performance of the instrument. The instrument was then reconfigured for a new probe called the NMR-MOLE and further testing was performed. This probe was still undergoing development and had not been previously tested. Finally, a dedicated compact instrument measuring 360 x 240 x 55 mm and weighing 3.6 kg was developed for the NMR-MOUSE probe.

Acknowledgments

My project and the writing of this thesis has been a long and trying task and I am indebted to and very grateful for all those who have helped me during this process.

I would like to thank my supervisors Professor Paul Callaghan and associate Professor Robert O'Driscoll for their guidance, encouragement, ideas and support. It has been a privilege to work with such competent colleagues and I have been inspired by the passion they have for their discipline.

I also would like to thank the staff within the Institute of Fundamental sciences. I am very grateful to all those who have helped me in various matters and for delivering on my sometimes demanding requests. In particular I would like to thank Terry Southern, Rob Ward and the staff within the electronics and mechanical workshops.

I am also grateful to Mark Hunter, Craig Eccles and Michael Adams for their ideas and assistance.

The people I would like to thank the most are my wife Christine and our children, Kerry, Haidee, Hannah and Cameron. Their support, patience and love was a tremendous help and I am deeply indebted.

Contents

Abstract i

Acknowledgments ii

Contents iii

1 Introduction 1

2 Nuclear Magnetic Resonance 4

2.1 Introduction to Nuclear Magnetic Resonance 4

2.2 Relaxation and the Bloch equation 9

2.3 NMR spectroscopy and bandwidth considerations 10

2.4 The Spin-Echo and CPMG experiments 13

2.5 NMR capabilities 15

2.6 NMR instrumentation 16

3 Portable NMR system design 19

3.1 Introduction 19

3.2 Review of portable NMR approaches 21

3.2.1 Earth's field NMR 21

3.2.2 Well logging probes 25

3.2.3 The NMR-MOUSE 29

3.2.4 The NMR-MOLE 31

3.2.5 Halbach magnet arrays 34

3.3 Outline of a portable NMR system design 35

4 System core 39

4.1 Introduction 39

4.2 System core hardware 44

4.2.1 Motorola DSP56309 and evaluation board 44

4.2.2 USB1.1 51
4.2.3 DSP/USB PCB design 57
4.3 System core software 63
4.3.1 DSP/USB board software 63
4.3.2 Windows 2000/XP device driver and API 67
4.3.3 Prospa 77

5 Digital transceiver **79**
5.1 Introduction 79
5.2 Digital transceiver hardware design 84
5.2.1 Transceiver design criteria 84
5.2.2 Transceiver devices 84
5.2.3 Transceiver circuit design and layout 94
5.3 Digital transceiver setup and software 104
5.4 Transceiver testing 109
5.4.1 Transmit section testing 109
5.4.2 Receive section testing 114

6 RF front end **117**
6.1 Introduction 117
6.2 Variable gain receiver amplifier 117
6.3 NMR-MOUSE RF front end 121
6.4 NMR-MOLE RF front end 122

7 System integration and testing **125**
7.1 Introduction 125
7.2 Hardware integration 127
7.2.1 Modspec 127
7.2.2 Lapspec 131
7.3 Software integration 134
7.3.1 Prospa experiment macros 135
7.3.2 Pulse programs 140

7.4 System testing 147

7.4.1 Probe tuning and matching 147

7.4.2 NMR-MOUSE spin-echo experiment 148

7.4.3 NMR-MOUSE CPMG experiment 150

7.4.4 NMR-MOLE spin-echo experiment 151

8 Conclusions 153

References 156

Appendix A Circuit schematics 163

Appendix A1 DSP/USB board schematics 163

Appendix A2 Transceiver board schematics 170

Appendix A3 Receiver amplifier board schematic 175

Appendix A4 Modspec power supply board schematic 177

Appendix A5 Modspec backplane board schematic 179

Appendix A6 Lapspec backplane board schematic 181

Appendix B Software 183

Appendix B1 DSP spin-echo pulse program 183

Appendix B2 Prospa spin-echo experiment macro 189

Appendix C Papers 193

Appendix C1 A DSP/USB platform for instrumentation and control 193

Appendix C2 Portable NMR systems for non destructive testing 200

Appendix C3 The Development of a Portable NMR System 207

Appendix D System specifications 214

CD containing a copy of the thesis, software and extra documentation.

1.0 Introduction

Nuclear Magnetic Resonance (NMR) is one of the more recent sensing technologies. It has become very popular due to its ability to non-invasively probe, down to the molecular level, the properties of many materials and living organisms. Although its greatest impact has been in the areas of chemistry and medical radiology, NMR is currently being applied to biochemistry, structural biology, and materials science research [1]. In the past ten years, NMR has made significant contributions to horticulture, biotechnology, chemical engineering, petroleum science and food technology and now stands on the threshold of making an impact on environmental monitoring, cultural heritage preservation, building technology, and security technology [2-6]. Traditionally NMR has been performed using laboratory or clinic based superconducting magnets, but now it is moving out into industry in the form of portable permanent magnet based systems. NMR development is being driven by advances in electronics, computing and magnet technology and so continues to advance in capability and application.

Superconducting magnets have been the key to the success of laboratory based NMR systems as they can produce homogeneous high strength fields up to 21 Tesla. However, superconducting magnets typically cost millions of dollars to purchase, are very expensive to maintain and require special facilities to house. Nevertheless the information that NMR can obtain makes them invaluable.

Nuclei are affected by other surrounding nuclei which gives rise to a distribution of resonant frequencies. This property is the heart of NMR spectroscopy and allows the determination of nuclear structure. Furthermore, by applying additional magnetic field gradients, spatial information can also be obtained. This is the basis for Magnetic Resonance Imaging (MRI).

The conventional wisdom used to be that a very homogeneous field is needed to do NMR. That has all changed. Recently, effort has gone into developing techniques to do NMR using the stray fields of superconducting magnets or the relatively inhomogeneous fields produced by simple permanent magnets [7,8]. These new permanent magnet based systems will be low cost, but limited in performance when compared with laboratory systems. Nevertheless they will still be very useful tools for specific applications.

The general philosophy behind portable NMR is to take the system to the sample rather than taking the sample to the system. This is of course necessary when one wants to make measurements outside the laboratory. Portable NMR technology has been and is continuing to be developed for applications such as the monitoring of rubber, polymers, timber, stone and many other materials. A trend in portable NMR is to design the complete system to perform only one specific task. This is very different from conventional NMR where the systems are designed to provide a wide range of experiments. Portable NMR systems should therefore be much simpler and cost less. The idea is that you use only what is needed to perform the task.

Since portable NMR tends to be application specific, the research is also more application driven compared with the more science driven research done on high field superconducting systems. In other words, a lot of the work is in solving industrial problems rather than doing fundamental science. Being more application focused means that portable NMR needs more of a multidisciplinary approach. This is highlighted in figure 1.0 where a total solution can be broken up into four main areas each involving different groups of people performing a particular function.

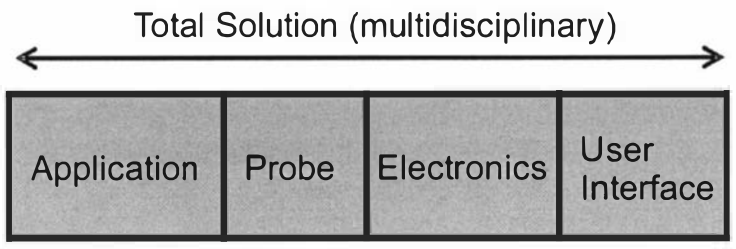


Figure 1.0 Development areas for a total solution.

Application developers are usually materials science researchers. Their function is to characterise the material of interest and determine what needs to be measured. Then a probe designer, familiar with magnets and electro-magnetics, is required to design a probe to suit the application. Next is the electronics instrumentation specialist who has to provide the supporting electronic system. Finally, the software programmer provides the user interface and the necessary data processing algorithms. However, the key to any NMR solution is a solid foundation of NMR expertise.

Portable NMR requires a different way of thinking and a more multidisciplinary approach. This is what is needed to take NMR to the next level in its development. Out of the laboratory and into the field! This new NMR technology vision will lead to a range of economical systems consisting of a permanent magnet based probe and an associated equipment case or backpack containing the necessary signal processing, control electronics and batteries. The ultimate goal is to make NMR based sensors that give a signal that is related to a material property such as, for example: flow, moisture, quality, softness or ripeness.

This work is part of the vision to develop a range of new NMR technologies in New Zealand. This vision was initiated by Professor Paul Callaghan and is now been carried on by a team of researchers from Massey University and Victoria University of Wellington. The research team is multidisciplinary with skills in NMR, physics, materials science, probe and magnet designing, software engineering, algorithm development and electronic instrumentation engineering. The combined effort of the team has resulted in some exciting technologies which are now marketed through a new company called "magritek" [9].

One of the new products is an Earth's field NMR system for teaching and research. This was a result of some earlier work where an Earth's field NMR system was developed for studying Antarctic sea ice [10]. This development has continued and has moved into the more challenging area of permanent magnet based portable NMR systems. The drive for this is to help solve some of the problems faced by industry. One of these problems is the determination of the moisture in concrete after it has been laid.

The goal of this work is to help make NMR a widely used technology and not limited, as now, to research and medical institutions. The key to this is the new electronics and computing technologies that continue to unlock new opportunities. The main task of this work has been to design and construct a complete NMR spectrometer that could be used with a range of probes that have been and are still being developed. The result is the electronics instrumentation component of a total solution. The intention has been to end up with a commercially viable system. Therefore, cost, ease of manufacture and user interface design were all critical factors. This thesis describes the development of two different NMR spectrometers: a modular user-adaptable version for NMR developers and a customised more compact spectrometer for NMR users. Both are based on a common set of modules. This work lays the foundation for future work where systems will be developed for industrial applications. It also provides a head start into the development of autonomous NMR based sensors. The work has also resulted in commercial products and this thesis will provide additional information for those who will work with those systems.

The following chapter presents a brief introduction to NMR, with some further information regarding the areas relating to this project. The third chapter reviews some existing portable NMR technologies and concludes with an evaluation of the common system requirements and an outline of the design. Chapter four gives the details of the design and construction of the "system core" module mentioned in chapter three. Chapter five describes the design and construction of a general purpose Radio Frequency (RF) transceiver module and chapter six describes the RF front end for two different probes. Chapter seven discusses the assembly of the hardware modules into enclosures, then deals with software/hardware integration and finally demonstrates the operation of the complete spectrometers. The final chapter presents conclusions and future directions.

2.0 Nuclear Magnetic Resonance

2.1 Introduction to Nuclear Magnetic Resonance [†]

Nuclear Magnetic Resonance (NMR) is a technique that allows one to non-invasively analyse the physical and chemical properties of materials by probing the nuclear dipole moments within the material. Nuclei of certain atoms have an intrinsic spin with angular momentum (L) that gives rise to a magnetic dipole moment (μ) and can be considered to be like a small spinning magnet. The relationship between the spin angular momentum and the magnetic moment is given as

$$\tilde{\mu} = \gamma\tilde{L} \tag{2.0}$$

where γ is called the *gyromagnetic ratio* and is nucleus-dependent. For protons $\gamma = 2.675 \times 10^8 \mathrm{s}^{-1}\mathrm{T}^{-1}$. When these nuclei are placed in a magnetic field (B_0), the magnetic dipole experiences a torque that causes them to align with the field. The torque is proportional to the applied field and can be expressed as:

$$\tilde{\tau} = \tilde{\mu} \times \tilde{B} \tag{2.1}$$

The torque changes the angular momentum of the nucleus and since the angular moment is parallel to the dipole moment the result is a differential equation

$$\frac{d\tilde{\mu}}{dt} = \gamma\tilde{\mu} \times \tilde{B} \tag{2.2}$$

Equation 2.2 describes a precessional motion of the magnetic moment about the applied magnetic field that is analogous to the motion of a spinning top in the Earth's gravitational field (figure 2.0).

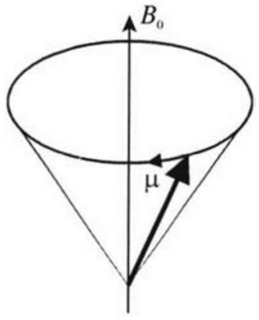


Figure 2.0 Motion of a magnetic dipole within a magnetic field.

The frequency of this precession is known as the resonant or Larmor frequency (ω_0) and depends on the type of nuclei and the strength of the applied magnetic field B_0. The equation for this relationship is

$$\omega_0 = \gamma B_0 \tag{2.3}$$

[†] Note that this is a simplified description of inherently quantum mechanical phenomena. For a more in depth description the reader is directed to a standard NMR text such as reference 1.

where again γ is the gyromagnetic ratio, which for hydrogen nuclei equates to a precession frequency of 42.58 MHz/Tesla. The hydrogen nucleus (proton) has a *spin quantum number* $I = \frac{1}{2}$ and therefore its *magnetic quantum number* (m_I) can take on the values $\pm\frac{1}{2}$. This results in two possible orientations for the Z component of the magnetic moment (μ_z) which corresponds to either of two states in a magnetic field, one with its magnetic moment aligned along the field direction and the other anti-aligned as shown in figure 2.1.

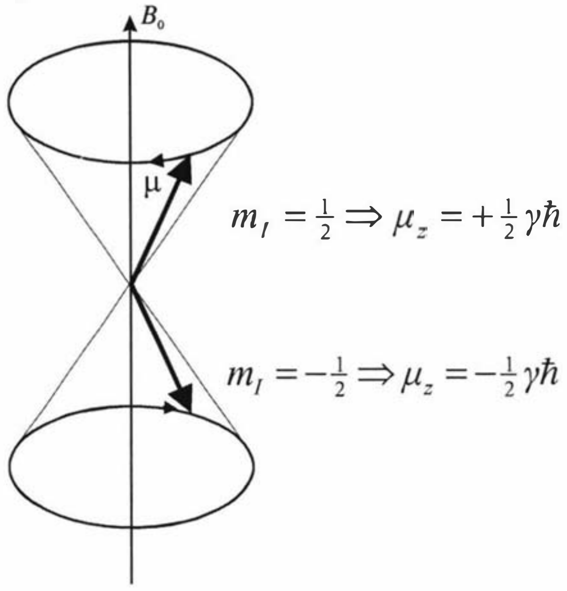

B_0
μ
$m_I = \frac{1}{2} \Rightarrow \mu_z = +\frac{1}{2}\gamma\hbar$
$m_I = -\frac{1}{2} \Rightarrow \mu_z = -\frac{1}{2}\gamma\hbar$

Figure 2.1 The two possible orientations for protons.

The two orientations have corresponding energies $\pm\frac{1}{2}\gamma\hbar B_0$ which results in an energy difference of $\gamma\hbar B_0$ as shown in figure 2.2.

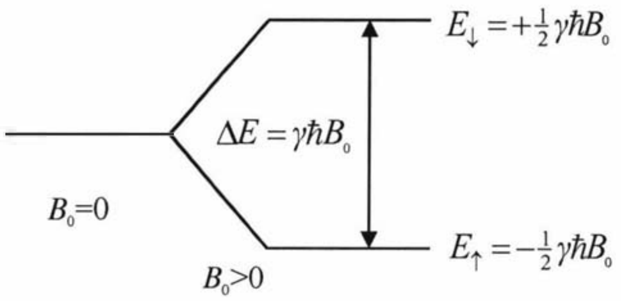

$E_\downarrow = +\frac{1}{2}\gamma\hbar B_0$
$\Delta E = \gamma\hbar B_0$
B_0=0
$E_\uparrow = -\frac{1}{2}\gamma\hbar B_0$
B_0>0

Figure 2.2 Energy levels for protons.

It is possible to make the nuclear spin jump from the lower state to the higher state by injecting a photon of the correct frequency

$$\Delta E = E_\downarrow - E_\uparrow = \gamma\hbar B_0 = \hbar\omega_0 \tag{2.4}$$

where again ω_0 is the Larmor frequency.

At room temperatures there is a slightly greater number of aligned nuclei than anti-aligned and the difference is given as

$$N_{\uparrow} - N_{\downarrow} = N_S \frac{\gamma \hbar B_0}{2kT} \tag{2.5}$$

where N_S is the total number of protons per unit volume (for water $N_S = 6.691 \times 10^{28}\,\mathrm{m}^{-3}$) and k is the Boltzmann constant. The phases of the spins are randomly distributed (figure 2.3) and so the sum of all the nuclear magnetic moments results in a small net magnetization $\widetilde{M}_0$ that is proportional to and in the direction of the applied field.

$$\widetilde{M}_0 = (N_{\uparrow} - N_{\downarrow})\mu_Z = (N_{\uparrow} - N_{\downarrow})\tfrac{1}{2}\gamma\hbar = \frac{N_S \gamma^2 \hbar^2 \widetilde{B}_0}{4kT} \tag{2.6}$$

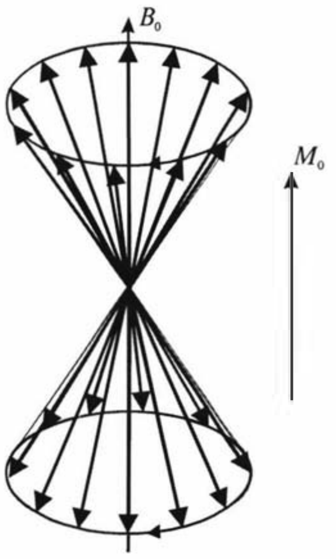


Figure 2.3 Distribution of nuclear magnetic moments.

In summary, when protons are placed in a magnetic field they will precess at a rate proportional to the field strength and they will either align with or against the field as shown in figure 2.4. There will also be a slightly greater number of aligned spins which results in a small net magnetization aligned with the field that can now be represented as a single vector as shown in figure 2.5. The magnitude of this vector is also proportional to the field strength.

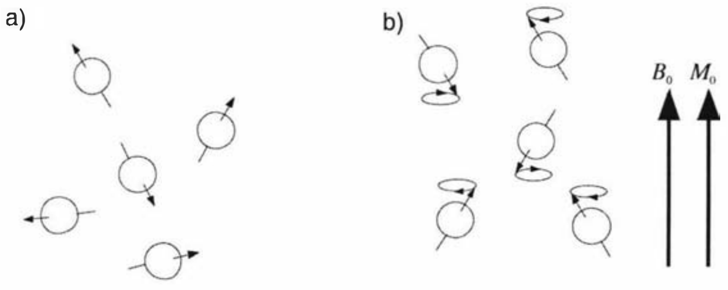


Figure 2.4 Nuclear magnetic moments before (a) and after (b) the application of a magnetic field.

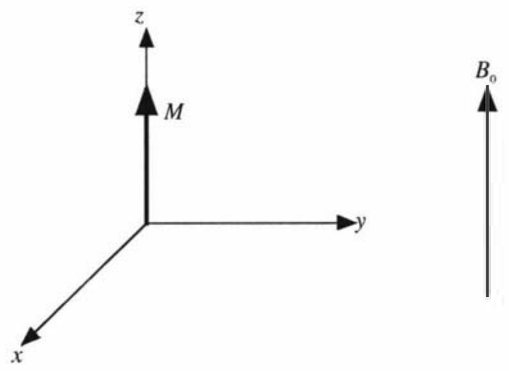


Figure 2.5 Net equilibrium magnetization M aligned with the applied field B_0.

It is possible to measure this magnetization by firstly causing it to rotate and then detecting its subsequent precession using a pick-up coil. To rotate the magnetization a torque needs to be applied. This is done by applying another magnetic field. The problem is that it would normally require a static field significantly larger than B_0 to do this. The way around this is to move into a frame of reference rotating at the Larmor frequency. The nuclear magnetic moments then become stationary ($\omega_0 = 0$) and B_0 disappears as $\omega_0 = \gamma B_0$. If a magnetic field is applied that is rotating at the Larmor frequency it will appear in the rotating frame as a stationary field and so is able to apply a torque to the apparently stationary magnetization. The field used to do this is called the B_1 field and is produced by driving a coil with a sinusoidal RF signal. The coil usually surrounds the sample and is orientated in such a way so that the magnetic field it produces is at right angles to B_0.

The oscillating magnetic field generated by the coil can be represented as two counter rotating field vectors. One of the vectors rotates in the opposite direction to the spin precession and so can be ignored as it would have negligible effects on the spin system and so the oscillating magnetic field can be represented as a single rotating field vector (B_1) that is at right angles to B_0 (figure 2.6). The result of all this is that only a small magnetic field is required to interact with the spin system as long as it is synchronised to it (or "at resonance").

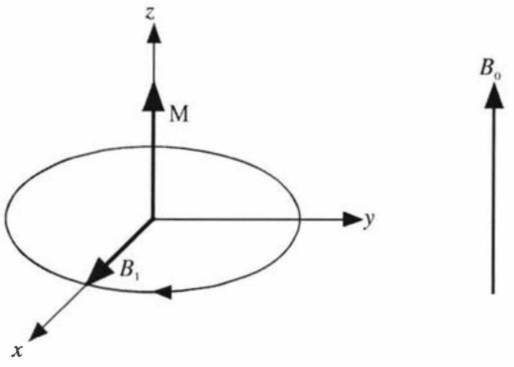


Figure 2.6 Rotating B_1 vector produced by RF pulse.

The torque applied by the rotating field B_1 causes the magnetization vector M to spiral down around the z axis (figure 2.7). When it reaches the x,y plane the RF field is removed and the precessing magnetization can then be observed as an induced EMF in a coil surrounding the sample. The B_1 field pulse used to achieve this is known as a *90-degree* pulse as it tips the magnetization by 90° into the x,y plane.

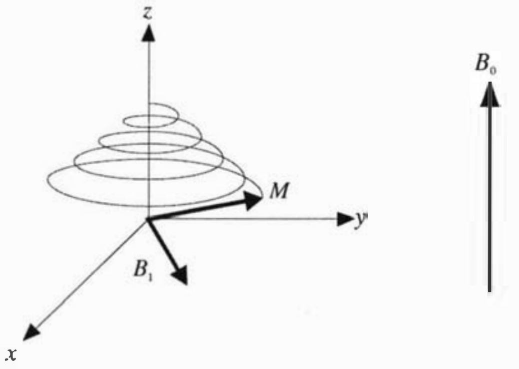


Figure 2.7 Magnetization vector M spirals down into the x,y plane due to applied B_1 pulse. The precessing magnetization is then observed as an induced EMF in a coil surrounding the sample. Often the coil used to apply the RF pulse is used to receive the induced signal.

The strength of the signal induced in the coil from the spinning magnetization can be derived using Faraday's law, where the EMF induced in a wire loop is proportional to the rate of change of the magnetic flux within the loop. This relationship written in a more useful form [11] is given as

$$e = -\frac{d}{dt}\left(\frac{\tilde{B}_1}{I}\cdot\tilde{m}\right) \qquad (2.7)$$

where B_1/I is the B_1 field due to unit current and $\tilde{m}$ is the net magnetic moment. The magnitude of the peak voltage for a sinusoidally varying magnetization can then be calculated using

$$e_s = \frac{B_1}{I} M_0 V_s \omega_0 \qquad (2.8)$$

where M_0 is the sample magnetization, V_s the sample volume, and ω_0 the precessing rate. The peak voltage e_s is then determined by the ability of B_1 coil to couple this alternating field. The coupling performance is gauged using the principle of reciprocity [12], where the ability of the coil to receive is directly proportional to its ability to generate a magnetic field. Note that the sample magnetization and the precessing rate are both proportional to the B_0 field. This is a reason why there is the continual demand for stronger super-conducting magnets. The induced signal does not last forever as relaxation processes will cause the magnetization to de-phase and also return to its equilibrium position. The resultant signal is called the Free Induction Decay (FID) and consists of a combination of decaying sinusoids (figure 2.8).

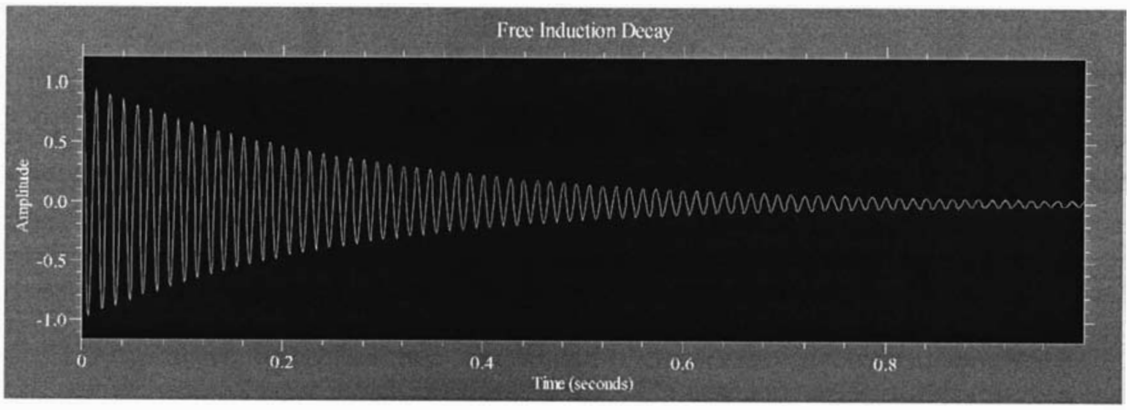


Figure 2.8 A simulated Free Induction Decay consisting of a single decaying sinusoid. The received signal is usually mixed with a frequency reference resulting in a lower frequency signal that can easily be acquired. In this example the difference between the two is 75 Hz.

2.2 Relaxation and the Bloch equation

The random thermal motions of the surrounding molecular environment of the spins (known as the lattice) produce localised field fluctuations that cause the magnetization to be restored to the B_0 field direction. Fluctuations at the Larmor frequency can cause energy to be transferred between the spin system and the lattice. The rate at which these fluctuations occur is lattice dependent and so the time taken to restore the magnetization to its equilibrium position provides information about the molecular environment. This restoration mechanism is called *longitudinal relaxation* or *spin-lattice relaxation* and is described by the function

$$M_z = M_0\left(1 - e^{-t/T_1}\right) \tag{2.9}$$

where T_1 is the characteristic relaxation time and for liquids is typically between 0.1 and 1 s. The second relaxation process is called the *spin-spin* relaxation or *transverse* relaxation and is caused by static spin-spin interactions or inhomogeneities in the magnetic field. This results in the spins precessing at slightly different frequencies and therefore getting out of phase with each other. This is shown in figure 2.9 where eventually the spin phases are distributed evenly in the transverse plane resulting in zero net transverse magnetization.

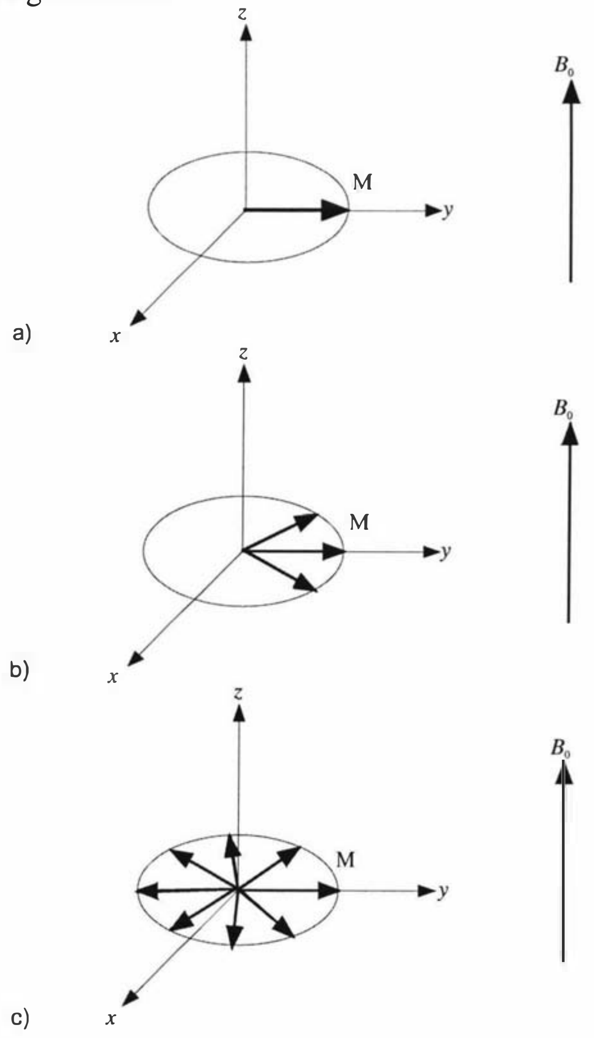


Figure 2.9 Dephasing of the magnetization due to variations in the B_0 field.

The B_0 inhomogeneities are caused by imperfections in the applied B_0 reference field and also by the localised magnetic effects of the other surrounding nuclei. The effect of the surrounding nuclei is much greater in solids than in liquids. The signal duration from a solid is in the order of a few microseconds whereas the signal from liquid water can last several seconds. The equation that describes this process is

$$M_{xy} = M_0 e^{-t/T_2} \tag{2.10}$$

where T_2 is the exponential decay time constant that dictates the envelope of the FID. If the B_0 reference field is not homogeneous then it can become the major contributor to the T_2 relaxation and so the term T_2^* is used to distinguish between the actual T_2 of the material and the decay rate of the FID. There is a method of determining the true T_2 value of a material which will be described shortly.

The FID can now be described by the *Bloch equations*:

$$\frac{d\tilde{M}}{dt} = \gamma\tilde{M} \times \tilde{B} - \frac{M_x\hat{i} + M_y\hat{j}}{T_2} - \frac{(M_z - M_0)\hat{k}}{T_1} \tag{2.11}$$

Note that this is similar to equation 2.2, but we now use the bulk magnetisation vector $\tilde{M}$ and include the relaxation processes T_1 and T_2.

2.3 NMR spectroscopy and bandwidth considerations

With NMR, the strength of the emission tells us about the number of nuclei present in the sample and is proportional to the sample volume when the nuclear density is uniform. The emission spectrum tells us about the magnetic environment of the nuclei. Any variations in the local field will cause the nuclei to precess with a slight frequency offset. This is often due to the magnetic shielding of the surrounding electrons and is referred to as the *chemical shift*. The offset is usually expressed as parts per million and is calculated using

$$\delta_i = \frac{\omega_s - \omega_0}{\omega_0} \times 10^6 \tag{2.12}$$

where ω_s is the offset frequency. The chemical shift is dependent on the B_0 field strength. The stronger the B_0 field the greater the difference will be between ω_s and ω_0. Hence, the stronger the magnet the better the spectral resolution. Chemical shift forms the basis for NMR chemical spectroscopy where the spectrum can provide information about the type of nuclei and their bonding arrangements, thereby telling us something about the makeup of a chemical sample (figure 2.10).

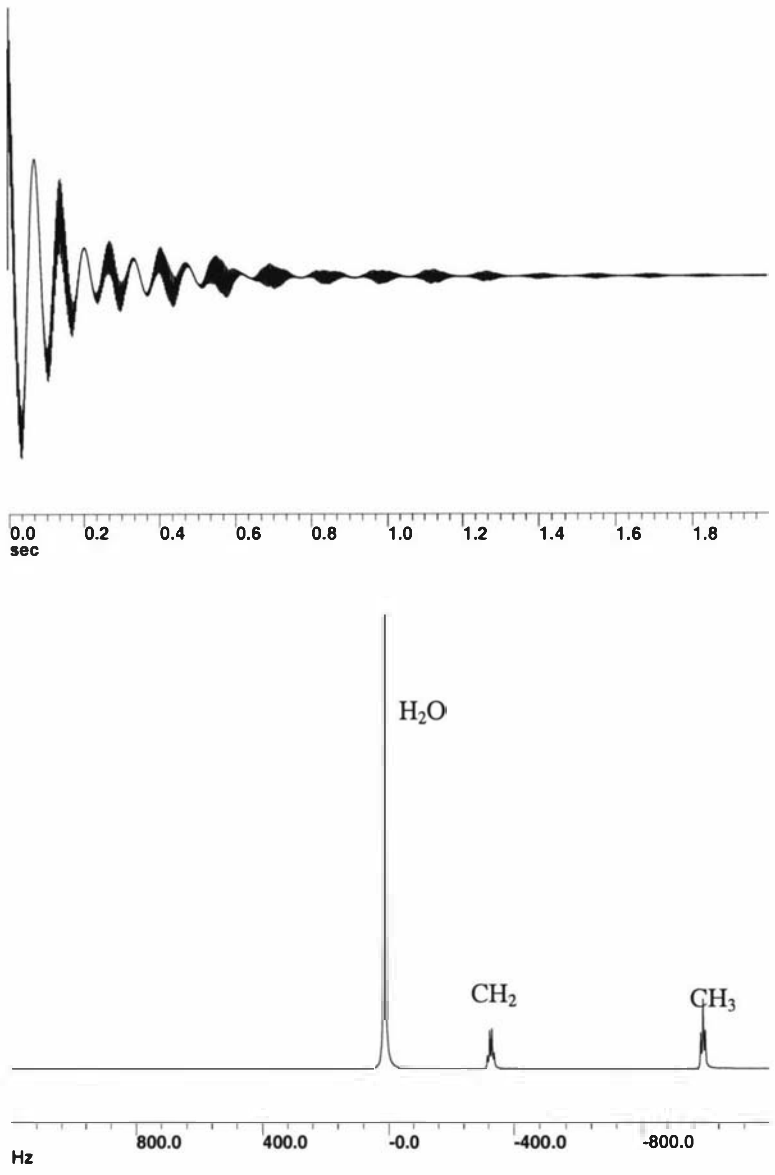


Figure 2.10 The FID and corresponding spectrum for a Tequila sample acquired with a 270 MHz NMR spectrometer [13]. The dominant decaying sinusoid and the corresponding dominant peak in the spectrum are from the protons contained within the water. The protons within the CH_2 and CH_3 components of the alcohol have a different magnetic environment and therefore precess at slightly different frequencies as indicated.

The most basic type of pulsed NMR experiment is the simple pulse sequence consisting of a single 90° pulse and the observed FID. This is all that is required to get a spectrum or to determine the water content within a sample (figure 2.11).

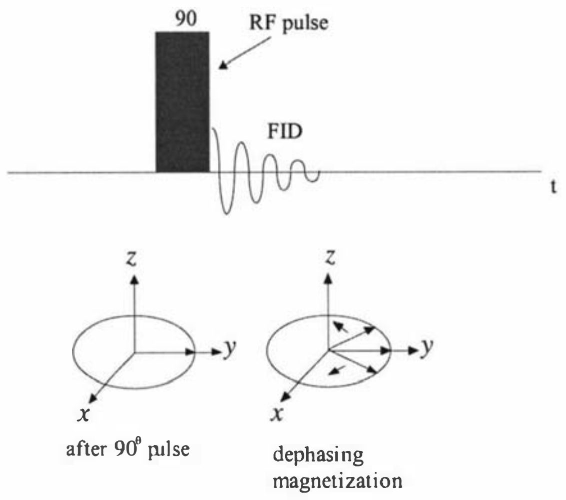


Figure 2.11 Simple 90° pulse sequence.

One consideration when designing an experiment is the amount of power required and the duration of the RF pulses. For an inhomogeneous B_0 field there will exist a range of Larmor frequencies. The method used to excite this band is to use a sufficiently narrow B_1 RF pulse. The excited bandwidth is proportional to the reciprocal of the pulse duration, this relationship being based on the well known Fourier transform property, where a narrow pulse in the time domain results in a broad spectrum in the frequency domain. If the RF pulse is very narrow, the power level required will be very high as a certain amount of energy is required to tip the magnetisation. The B_1 field strength and the magnetisation tipping rate (ω_1) are related by

$$\omega_1 = \gamma B_1 \tag{2.13}$$

where again γ is the gyromagnetic ratio. For a 90° (π/2) pulse we can derive the following relationship

$$\vartheta_{tip} = \frac{\pi}{2} = \omega_1 t_{90} = \gamma B_1 t_{90} \Rightarrow B_1 t_{90} = \frac{\pi}{2\gamma} \tag{2.14}$$

where $B_1 t_{90}$ is constant and is the product of the strength and duration of the pulse.

Another consideration when designing an experiment is that NMR is a relatively insensitive technique and therefore to improve the signal to noise an experiment will often need to be repeated and the data averaged. In order for the resultant signals to co-add correctly it is therefore necessary that the experiment is executed repeatedly in exactly the same way. This requires the hardware to have precise timing capabilities.

2.4 The Spin-Echo and CPMG experiments

In an NMR experiment the decay of the FID is largely due to the dephasing of the magnetization, caused by the nuclei precessing at slightly different rates at different positions in the sample. If another RF pulse is applied a short time after the FID, so that the magnetization components are flipped over by 180 degrees, a "spin-echo" will form [14] due to the rephasing of the magnetization (figure 2.12). After reaching a maximum the echo will then start dephasing as before.

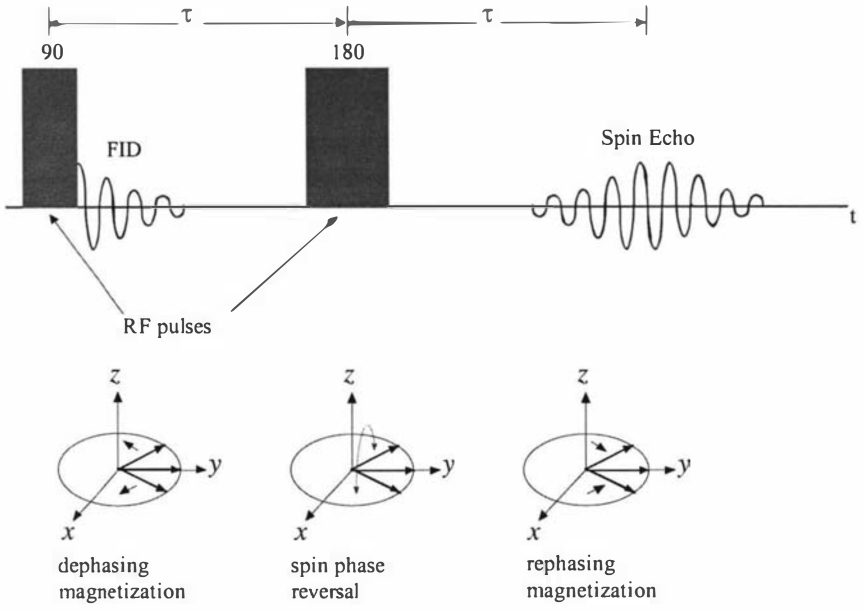


Figure 2.12 Spin-echo formation due to the refocussing of the magnetization.

Not all of the spins will refocus perfectly as some will experience slight changes in their magnetic environment during the time between the 90° pulse and the spin-echo formation. Apart from relaxation processes, the main contributor to partial refocusing is the movement of the nuclei to different positions in an inhomogeneous B_0 field, and hence a change in their magnetic environment. This leads to a slight attenuation of the spin-echo that is proportional to the motion of the nuclei. We can utilise and enhance this effect by applying additional magnetic field gradients and therefore make the echo attenuation much more sensitive to nuclear movement. This forms the basis for diffusion measurements. When working with very inhomogeneous B_0 fields the spin-echo experiment is essential as often T_2^* is shorter than the ring-down time of the B_1 coil thereby making it impossible to detect a FID.

After the first echo is detected one can apply another 180° pulse which will then cause another echo to be formed. This can be repeated so that a whole series of echos can be generated. A specific example of this technique is the Carr-Purcell-Meiboom-Gill (CPMG) sequence [15,16] where an initial 90° pulse is followed by a series of 180° pulses with the carrier phases shifted 90° relative to the first pulse. The maxima of the echo amplitudes will follow a T_2 decay envelope as shown in figure 2.13.

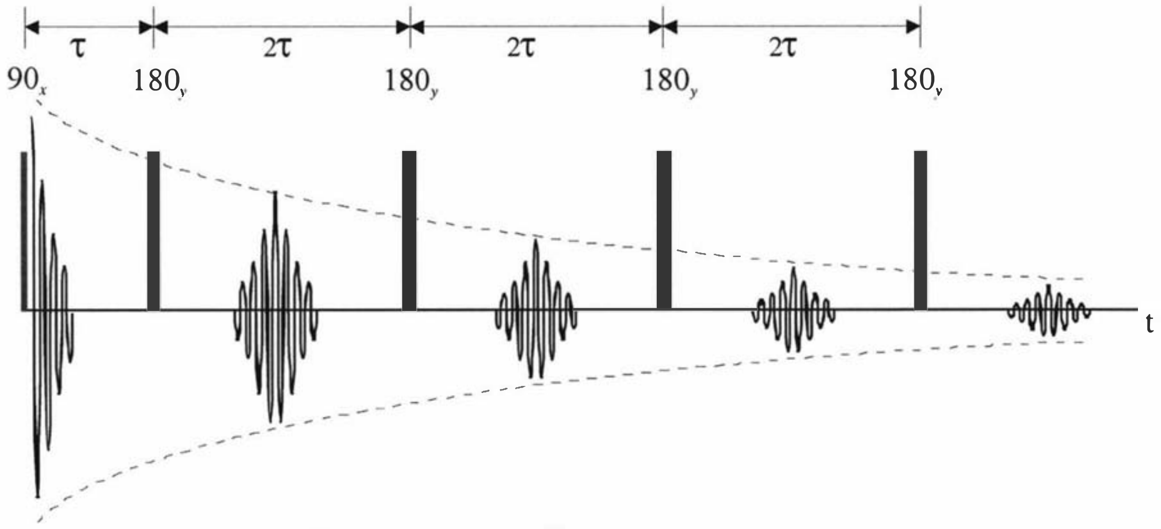


Figure 2.13 CPMG spin-echo formation with T_2 decay envelope.

The CPMG method provides a one shot determination of T_2 and has become very useful especially when the field inhomogeneity is such that $T_2^* << T_2$. An example of this is shown in figure 2.14 where an inhomogeneous permanent magnet based probe was used to obtain data from a piece of rubber. The individual spin-echoes lasted about $10\,\mu s$ (T_2^*) and so are very narrow when compared to the overall T_2 decay curve.

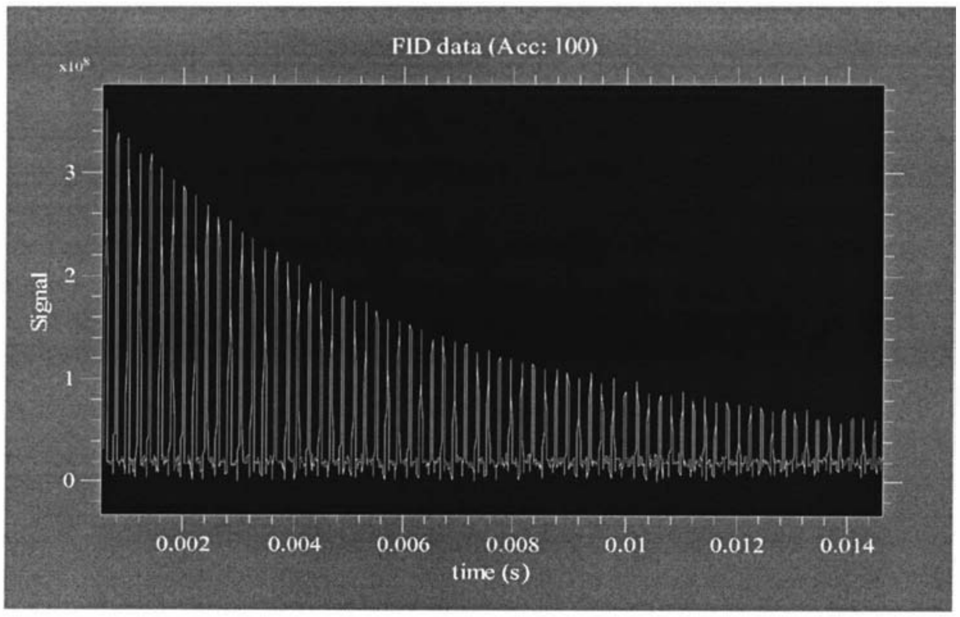


Figure 2.14 CPMG data of a rubber sample in an inhomogeneous B_0 field.

2.5 NMR capabilities

Pulsed NMR is in some ways similar to the impulse response techniques used in electronic systems, in which a pulse is used to excite a system and the system's response or emission reveals some properties of the system (Figure 2.15).

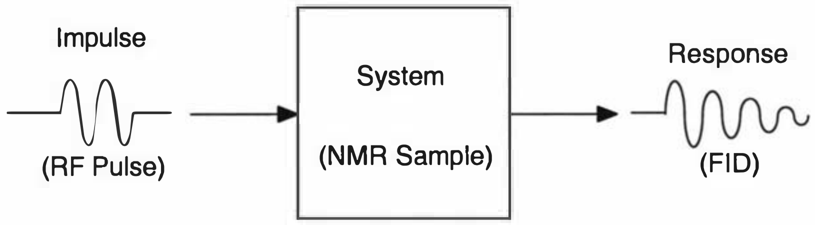


Figure 2.15 The NMR impulse response.

NMR has become a standard tool for chemical spectroscopists as it has greatly enhanced their ability to analyse the chemical make-up of solutions and the determination of the structure and environments of various molecules. Another very important application of NMR is imaging. Magnetic field gradients are applied to make the Larmor frequency position dependent, and therefore localised information can be obtained. This information can then be used to construct images. This technique, known as Magnetic Resonance Imaging (MRI), has become extremely popular in the medical profession, but is also very useful in materials science research. By combining the imaging technique with some other techniques, such as diffusion, it is possible to get weighted images, where the intensity is proportional to a parameter of interest. Some of the weighted imaging techniques [11] that can be performed are:

- Nuclear density.
- Diffusion and velocity (flow) imaging.
- Relaxation imaging.
- Chemically selective imaging.

A major advantage of MRI over other imaging techniques is that it is largely non-invasive. It does not usually require any physical alteration of the sample, unlike optical techniques, and it does not require the use of harmful radiation, such as x-rays or the emissions of radioactive substances. When spatial localisation is not required the NMR technique is capable of making averages over the sample volume which with other techniques might be more time consuming and difficult. NMR with its ability to analyse the physical and chemical properties of materials, has become a very important research tool while its variant, MRI, has made a major impact as a medical diagnostic tool.

2.6 NMR instrumentation

To perform NMR experiments you need a magnet, a coil and some electronic instrumentation to provide the stimulus and to subsequently detect the FID signal. A system that has all this is often referred to as a *NMR spectrometer*. A typical spectrometer consists of three basic sections as shown below in figure 2.16.

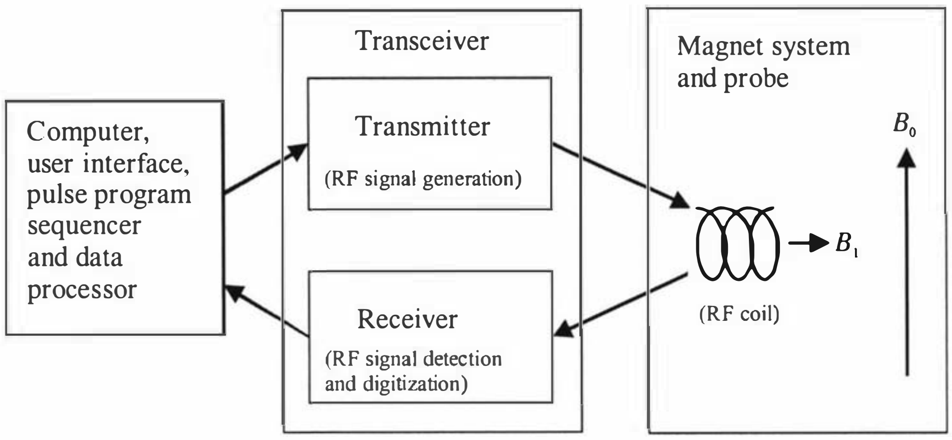

Computer, user interface, pulse program sequencer and data processor
Transceiver
Transmitter
(RF signal generation)
Receiver
(RF signal detection and digitization)
Magnet system and probe
B_0
B_1
(RF coil)

Figure 2.16 NMR spectrometer block diagram

The first section is usually based around a desktop computer and some other digital hardware. It performs the tasks of providing the user interface to the spectrometer, controlling all the sequencing of events and the processing and storage of the captured data. It is often considered to be the *digital* part of the spectrometer. The next section is often referred to as the *transceiver* or the *analogue* part as it deals with the generation and detection of the analogue signals and is very much like an Amplitude Modulation (AM) radio transceiver. The transmitter subsection provides the RF stimulus to the B_1 coil. The receiver subsection first amplifies the small RF FID signal. It then uses frequency mixing to shift the spectral information to audio frequencies and finally it digitises the audio signal and transfers it to the computer. The last section, which is often the most difficult and expensive section, is the magnet and probe. The stronger the B_0 field is, the better the sensitivity and spectral resolution. This has resulted in the push for stronger and stronger magnets with the cost going up significantly with field strength.

One of the most common types of magnet and probe geometries is the laboratory based vertically orientated superconducting magnet that provides the strong, stable and homogeneous fields that are necessary for high resolution spectroscopy. An example of one of these magnets together with the supporting electronics console is shown in figure 2.17. High resolution NMR spectroscopy has made a major impact within the chemistry discipline and is now a very important tool for protein structure determination.

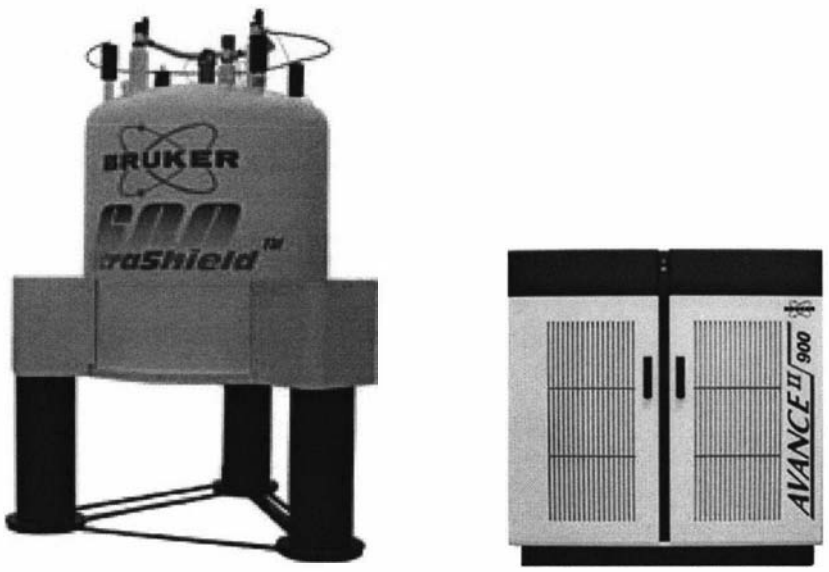


Figure 2.17 Superconducting magnet based NMR spectrometer for chemical spectroscopy. Figures taken from reference 17.

MRI systems have the extra requirement of field gradient generation and so additional hardware is required. One example of a system that is used for medical purposes is shown in figure 2.18. The field gradients are produced by driving coils with currents in the order of 10 to 100 A. The currents are switched on and off and at different levels and so a gradient control unit is required to generate the appropriate control signals. High power audio amplifiers are used to provide the large currents. MRI superconducting magnets are designed to cater for large samples, but the homogeneity requirements of the B_0 field are not as stringent as for spectroscopy systems. These systems have become a significant tool for medical diagnostics, but they are expensive, often costing several millions of dollars.

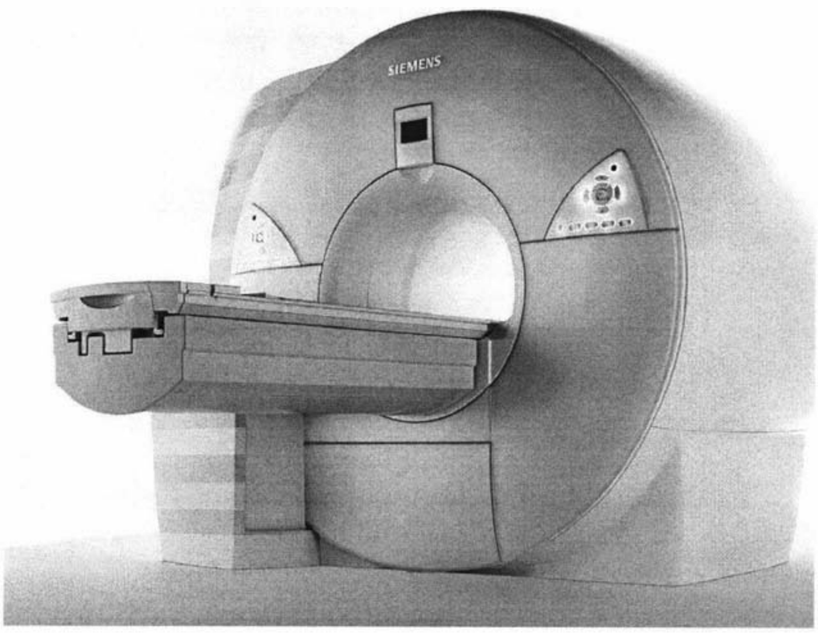


Figure 2.18 1.5 T superconducting magnet based whole body MRI system. Figure taken from reference 18.

Most imaging systems are whole-body systems and are used for medical purposes. However a few research laboratory systems exist such as the one shown in figure 2.19. This system is only capable of working with samples up to 90 mm in diameter, but it operates at 4.7 T which is significantly stronger than whole-body medical systems. These laboratory systems are often used for imaging plants or small animals and/or materials science research. As well as imaging they are also useful for determining material parameters such as the T_1, T_2, diffusion coefficient or moisture content.

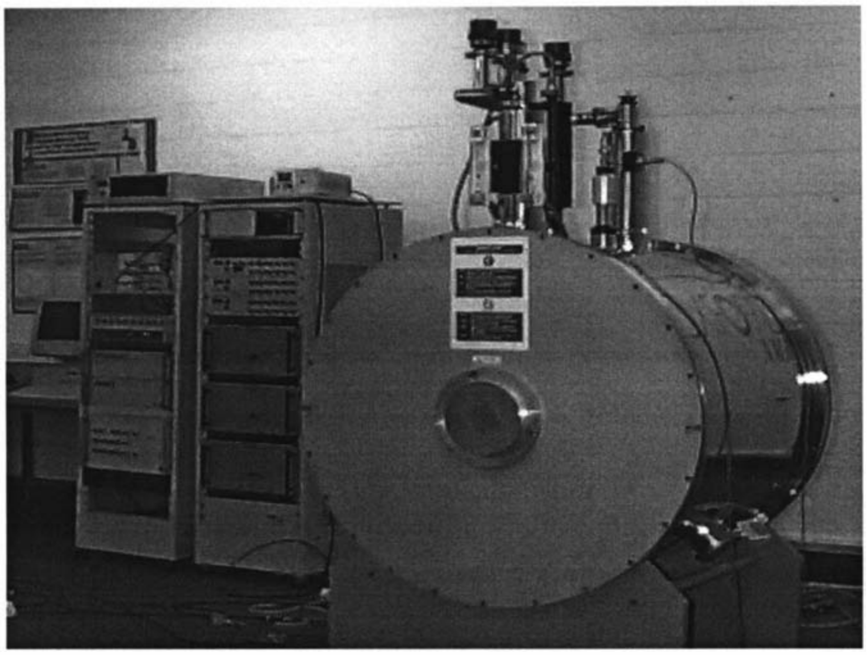


Figure 2.19 MRI system consisting of a 4.7 T superconducting horizontal magnet and two partially locally constructed instrument racks [2].

From an electronics instrumentation point of view, NMR/MRI is an interesting application area because it is technologically challenging. It tends to push all the technologies to their limits. The FID signal is very weak and there is often very little spectral separation between the various frequency components. What is required is very sensitive spectrometer electronics with very good long term frequency stability. In particular, the hardware designer must produce: high power RF amplifiers, very fast low loss duplexers, very low noise fast recovery preamps, fast high power audio amplifiers for gradients, wide bandwidth high dynamic range digitisers and low drift low phase noise oscillators. Even more challenging is the need to refine and repackage existing technologies to produce compact portable systems.

3.0 Portable NMR system design

3.1 Introduction

NMR systems come in various degrees of portability. Some systems require a truck, others a suitcase. If it is movable then it is considered portable. This idea has come about to distinguish portable systems from superconducting magnet systems which are considered immovable. The meaning of portable is defined by each user. However the definition of portable which is useful for this work is *a system that can be carried and operated by a single user*. The term "mobile" is often used, but again the meaning depends on the user. Portable NMR systems are application specific with the probe and supporting electronics being made specifically for a particular application.

The probe is the sensor. It is application specific and therefore comes in a range of sizes and designs. The probe is the interface between the material to be observed and the user and is critical to the performance of the whole system. All portable NMR systems, apart from Earth's field NMR, use permanent magnets to generate the required B_0 field. They are often open magnet designs where the probe can be placed against a surface. This is usually referred to as "inside-out NMR" to distinguish it from solenoidal geometries where the sample is enclosed within a coil or magnet array. Open magnet designs usually have much less field producing material in close proximity to the sample region and so the magnet designer is severely restricted in the field strength and homogeneity that can be achieved. Individual magnets are not ideal as they are non-uniformly magnetised. The magnetic material is also granular and the field strength is temperature dependent. Open magnet designs require the B_1 coil to be an open structure such as a surface coil. This results in poor B_1 field coupling in addition to the inhomogeneous and weak B_0 field. Therefore, inside-out probes usually suffer from poor signal to noise performance and require greater RF excitation power. Some surface probes are extremely inhomogeneous with field gradients in the order of $10\,\mathrm{Tm^{-1}}$. This means that it is virtually impossible to obtain an FID as the signal will have disappeared before the B_1 coil of the probe had finished ringing down. Therefore spin echo techniques must be used to obtain any signals.

At present there are no commercially available, compact, general purpose, portable NMR spectrometers. Some researchers have adapted existing bench-top or rack-mount systems to make them semi-portable, but they are still rather large (figure 3.0). This is because portable NMR is still in its infancy and has not really been applied to a specific industry problem. However with the advances that are occurring in the electronics and computing sectors, it is becoming much easier to design more compact systems. Already there has been a major increase in activity in this area and in the near future we should see a burst of products coming onto the market. Some companies that are now trying to develop systems are Quantum Magnetics [19], Tecmag [20] and Stelar [21].

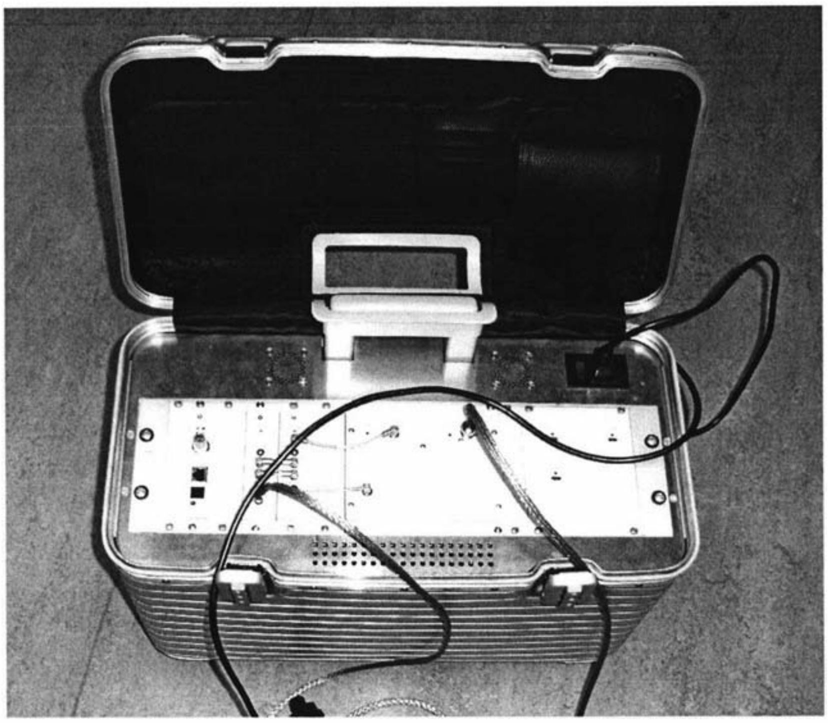


Figure 3.0 A Bruker Minispec NMR spectrometer mounted in a 19 inch trolley case, it measures 520 by 260 by 390 mm and weighs 11.5 kg. The preamp/duplexer unit is external and is not shown. Presently this is the smallest mobile NMR spectrometer in operation and is used with a series of permanent magnet based probes [22].

Quantum Magnetics has done a considerable amount of work in developing explosive detection systems based on the principles of Nuclear Quadrupole Resonance (NQR). This technique is very similar to NMR and so the electronic instrumentation requirements are almost the same. Tecmag mainly develop electronic consoles for superconducting magnet systems, but they are now moving into the area of portable NMR. Stellar produce field cycling NMR instruments which are large laboratory based systems, but like Tecmag they can see the future potential of the portable NMR market.

The goal of this work is to be able to put together a battery powered backpack system with a laptop computer as the user interface along with simple one-button operation software. It would then be taken to the sample to perform the measurements. This degree of portability will open up a whole range of new non-destructive application areas such as:

- Construction materials (timber, concrete)
- Elastomers (cross-link density, strain)
- Biological materials (tendons)
- Adhesive curing (car window bonding)
- Moisture in buildings (leaky buildings)

There are many difficulties in designing complete portable NMR systems. However significant progress has and is still being made in the area of portable NMR. Some of this progress will now be reviewed.

3.2 Review of portable NMR approaches

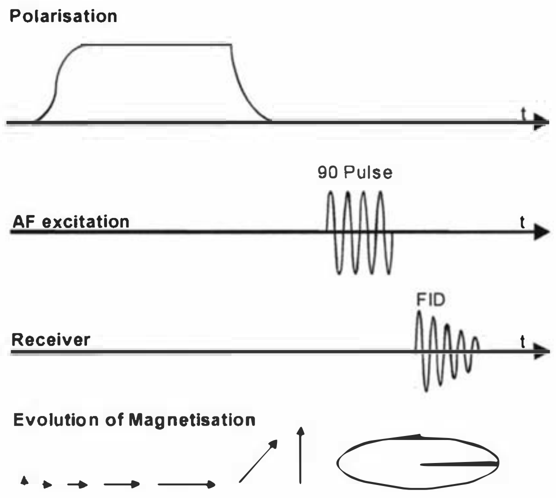
3.2.1 Earth's field NMR

Earth's field NMR makes use of the freely available magnetic field of the Earth to provide the B_0 field [23] and can therefore be inexpensive, easy to implement, highly portable, and able to be used in remote regions and under harsh environmental conditions, thereby enabling *in situ* measurements to be performed. One does not require a large expensive heavy magnet and a whole console full of electronics. Typically the field strength in New Zealand is about $56\,\mu\mathrm{T}$, which equates to a resonant frequency of 2.4 kHz. This field may be small, but it is very uniform, so FIDs lasting a few seconds can be obtained. Also, since the field is so uniform, large samples can be used, and therefore satisfactory signal levels can be obtained despite the low Larmor frequencies. As the frequency is in the audio range, quite simple transceiver circuitry is required, and the signal can be sampled directly, eliminating the need for Intermediate Frequency (IF) stages.

Earth's Field NMR has been successfully applied to a number of applications, one of these being the detection of subsurface water. One group [24] used two 17 m diameter three-turn loops configured in a figure 8 fashion to act as the B_1 coil. This was then placed on the surface of the earth directly above the area of interest. A high power pulse was applied to the coil to excite the nuclei within the subsurface water. The same coil was then used to receive the resultant FID. The field produced by the B_1 coil is depth dependent. This property was used to determine the depth of the water as the signal amplitude is dependent upon the excitation pulse amplitude and duration (known as the *pulse moment*). The greater the pulse moment the greater the depth. Typically pulses of up to 500 A and durations of up to 40 ms were used. This technique, like most NMR techniques, suffers from lack of sensitivity, but the sample volume is very large and compensates. A truck was used to transport the equipment to various sites and was necessary due to the high power demands of the system. This technique has been successful in detecting groundwater down to depths of 100 m and it has even been used to detect the presence of organic contaminants.

For smaller samples, such as a bottle of water, the sensitivity of the experiment can be further increased by the application of a polarization field at the start of the pulse sequence. This is applied to increase the sample magnetization and therefore increase the subsequently received signal strength. A solenoid is used to generate the polarization field (B_p), while a saddle coil or solenoid is used to generate and receive the perpendicular B_1 field (figure 3.1). The configuration used is determined by the sample orientation and access requirements. Generally, a solenoid is used whenever possible as better signal performance is obtained. The pulse sequences for Earth's field NMR experiments of this type are like conventional NMR sequences but with the addition of an initial 5 s polarizing pulse (figure 3.1). The solenoidal probe has its polarization coil oriented orthogonally to the Earth's field, and so the magnetization grows orthogonally to the Earth's field. But if the field is removed adiabatically[†] [25], the magnetization will re-orientate along the Earth's field direction.

[†] By adiabatically we mean that the polarization field is removed slowly enough that the magnetization is not left behind but follows.



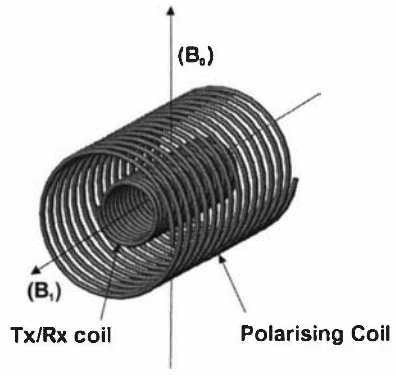


Figure 3.1 Simple pulse sequence and magnetization evolution for solenoidal probes.

In many ways the Earth's field NMR system is very similar to a conventional laboratory system but operates at audio frequencies instead of hundreds of MHz. An earlier system, that was part of some previous work, was built up from a number of interconnected units as shown in figure 3.2 [10]. The "RF" transceiver section which really operates at Audio Frequencies (AF) uses standard operational amplifiers and other audio components. The power required for the B_1 excitation is minimal, therefore a low noise preamplifier can be connected permanently to the probe. The inner block known as the "system core" is responsible for generating all the required signals and also for digitizing and processing the FID. The probe being the "transducer" end of the system consists of a single transmit/receive coil, a polarizing coil and a gradient coil. The rest of the components that make up the system are two constant current power supplies required for driving the polarizing and gradient coils and some other power supply electronics so that the whole system can be run from a 24 V battery.

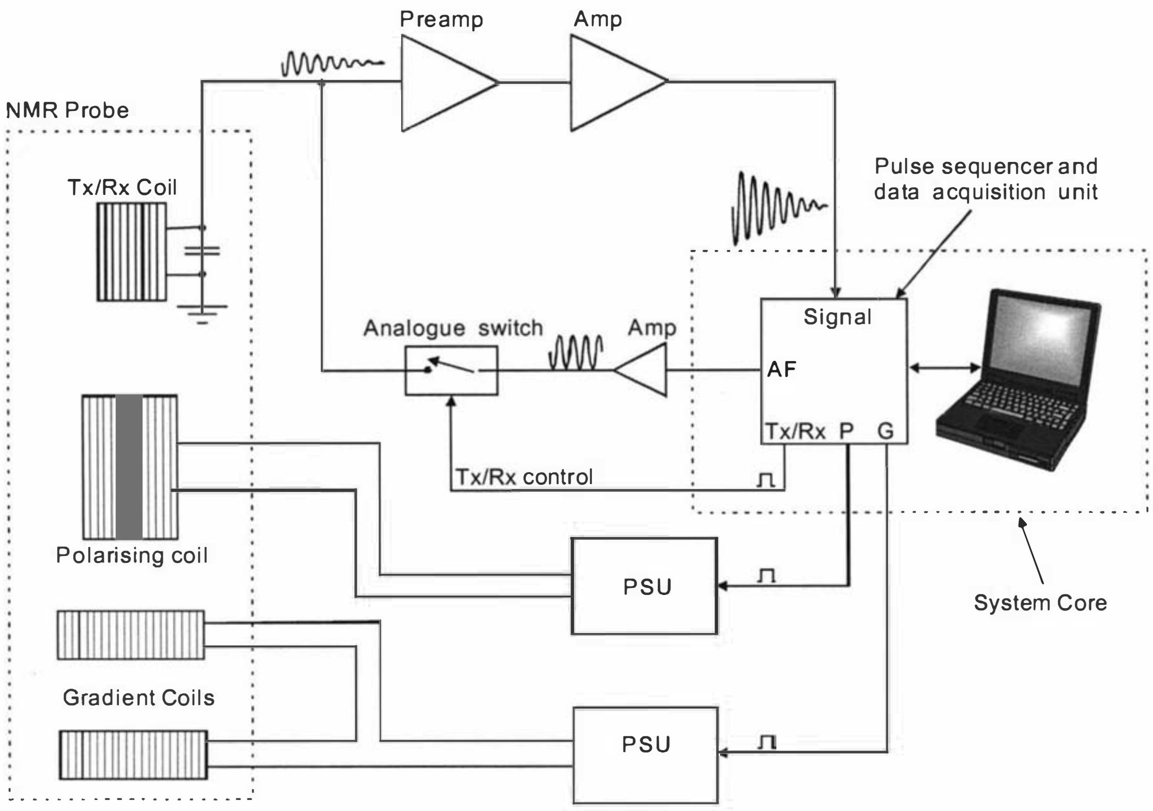


Figure 3.2 Block diagram of an Earth's field NMR system.

This system has been used in Antarctica [26, 27] to measure the abundance and mobility of brine trapped within the sea ice. A saddle coil was used for the B_1 coil so that the probe could be placed with its axis along the vertical position as shown in figures 3.3 and 3.4.

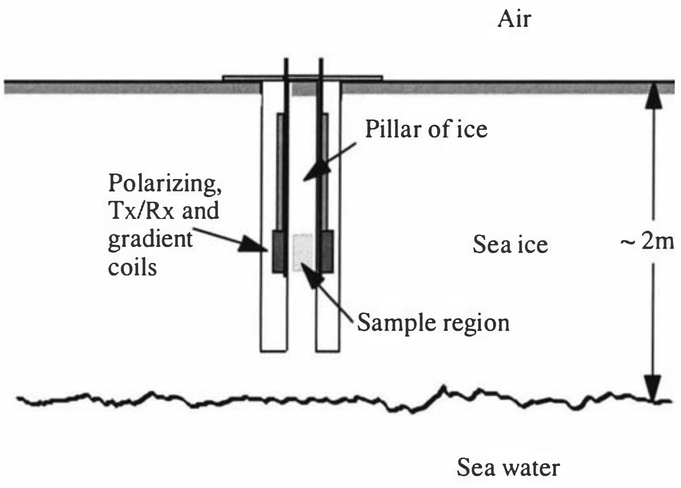


Figure 3.3 Sample and probe arrangement for sea ice work.

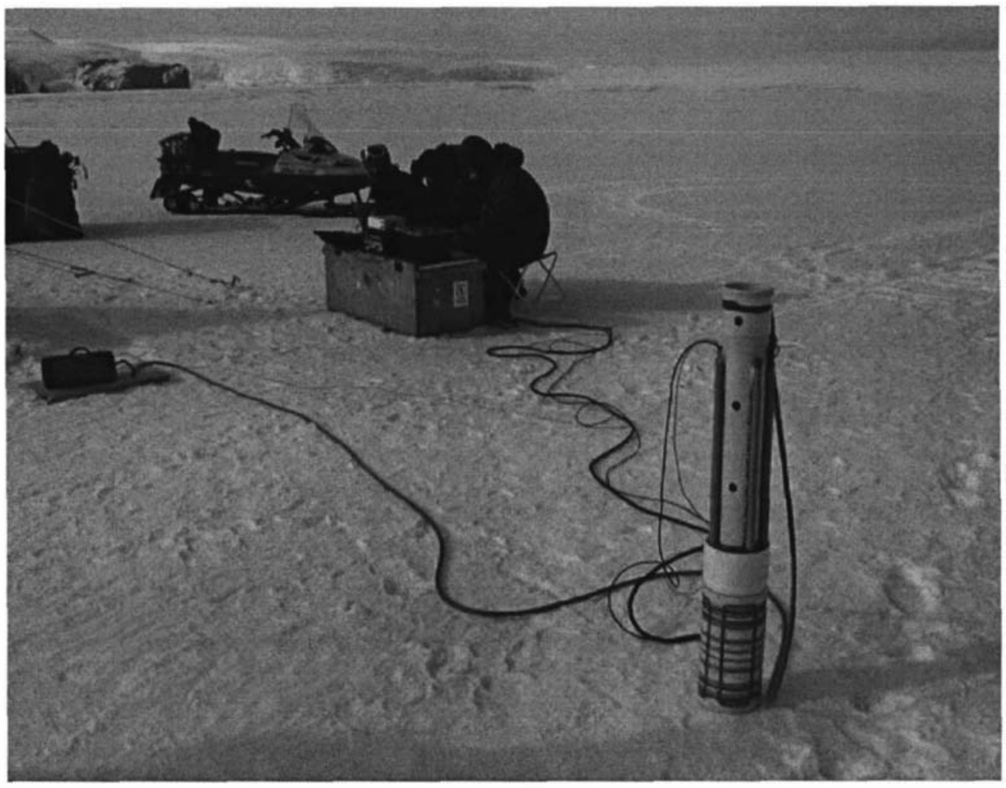


Figure 3.4 System in use (the probe is tested with a water sample before being inserted into the ice).

Earth's field NMR is quite novel and can be used for many different types of applications. Some of these applications are the same as for conventional high field NMR systems. Some examples are:

- Water content determination of soil, fruit, sea ice, etc.
- Self diffusion and flow of water [28].
- Relaxation studies of materials [29].
- Imaging of proton density, diffusion and relaxation weighted imaging [30, 31].

The low cost and versatility of Earth's field NMR makes it an ideal system for teaching. A commercial version of the earlier system has been produced so that others can utilise the versatility of this remarkable technology. This system is shown in figure 3.5 and is called "TerraNova" [9].

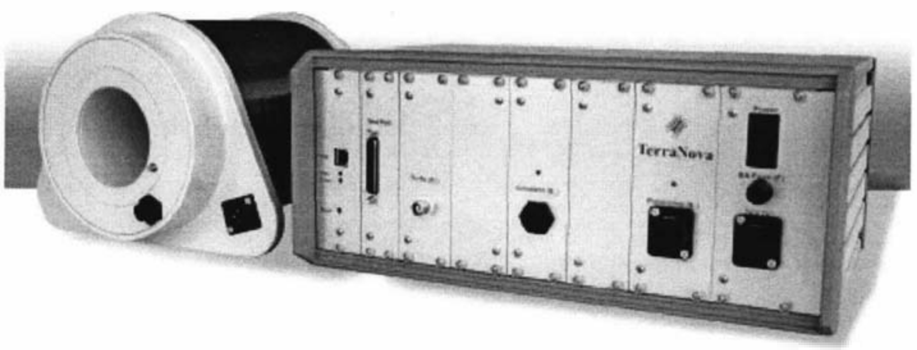


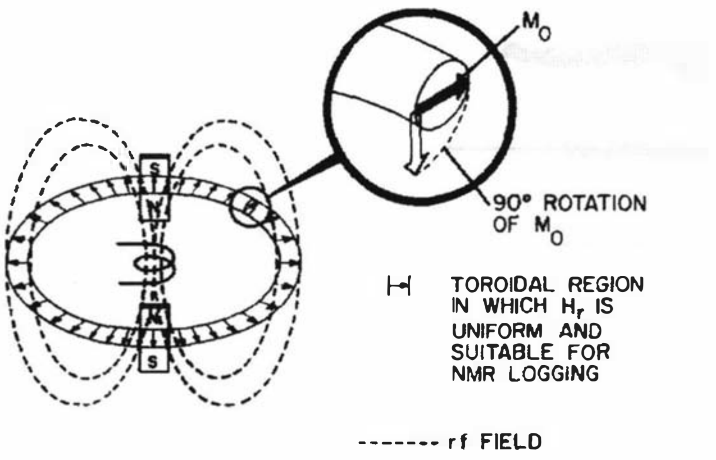
Figure 3.5 The TerraNova Earth's field NMR teaching and research system [9].

3.2.2 Well logging probes

NMR has become an important technology for the petroleum industries. Oil and gas is typically found in beds of porous rock as deep as 10km [32, 33]. The pores within the rock can be filled with oil, water and natural gas. NMR Well logging is concerned with measuring the properties of the porous rock to provide valuable information about the rock and the fluids contained within it. This is done by inserting a probe into a borehole and performing measurements as the probe is lowered or raised. Well logging is a good example of an application-specific NMR system that is used outside the lab. The sample is outside the probe which results in much reduced signal to noise performance. This, together with the borehole being a harsh environment with high temperatures and pressures, has made the application of NMR to this problem very challenging to the probe designers. However, the commercial value of the information that can be extracted has driven significant progress in this area for several decades.

The idea of using NMR to investigate borehole properties originated in the 1950's, not long after the NMR phenomenon was first discovered [34]. The first tool was developed in the early 1960's at the Chevron research laboratory [32, 33] and provided a capability that allowed a lot of foundational well logging research to be performed. It was an Earth's field based system that used prepolarising [35]. A single coil, oriented with its field direction orthogonally to the earth's field, was used to polarise the sample as well as to receive the FID signal. After the polarisation period the coil was rapidly switched off to leave the magnetisation aligned with the previous polarising field but precessing about the Earth's field. The precessing magnetisation was then detected using the same coil. This is in contrast to the adiabatic slow removal of the polarising field that causes the magnetisation to align with the Earth's field. Over the next 20 years the system design was further developed and adopted by other groups and was finally abandoned due to a new permanent magnet based tool developed by Jasper Jackson et al., at the Los Alamos National Laboratory [36, 37].

The new permanent magnet probe was developed in 1978 [33] and was based on the principle that if two cylindrical magnets are arranged with their same poles facing each other, a relatively uniform toroidally shaped magnetic field region would exist outside the magnet structure (figure 3.6). A solenoidal B_1 coil is placed between the magnets to generate the necessary perpendicular field. The main problems with this probe design are the weak field (500 kHz) and poor B_1 coupling. But this was pioneering work that inspired many other new designs and demonstrated the possibility of using permanent magnets for well logging. This design was dubbed "inside-out" NMR to distinguish it from the more common arrangement of the sample inserted within the magnet structure. One of the major advantages of using permanent magnets over an Earth's field system is the significantly reduced ring down time of the B_1 coil (deadtime). This feature allows the measurement of materials with shorter T_2 time constants.



M$_0$
90° ROTATION OF M$_0$
TOROIDAL REGION IN WHICH H$_r$ IS UNIFORM AND SUITABLE FOR NMR LOGGING
rf FIELD
S
N
N
S

Figure 3.6 The "inside-out" NMR device. Two magnets are arranged with similar poles facing each other. This generates an external toroidal magnetic field region. A solenoidal B_1 coil is used to excite the nuclei and receive the FID signal. Figure taken from reference 37.

An alternative design was developed in 1985 by a group from NUMAR Corporation who were originally trying to optimise the previous example [33, 38, 39]. This design uses a single cylindrical magnet magnetised perpendicular to the cylinder axis (figure 3.7). This produces a non-uniform field with a strength that is proportional to the distance from the magnet. The result is a resonance region shaped like a thin cylindrical shell. The B_1 coil is wound around the cylindrical magnet along the North and South pole faces. This provides a B_1 field that is perpendicular to the B_0 field for all of the surrounding space. This design is interesting in that it can be considered an inhomogeneous design, therefore departing from traditional homogeneous methods. The design has resulted in a commercial well logging device known as the MRIL tool (Magnetic Resonance Imaging Log). Later versions of the tool use multiple sensitive volumes at different resonant frequencies to increase the logging capacity.

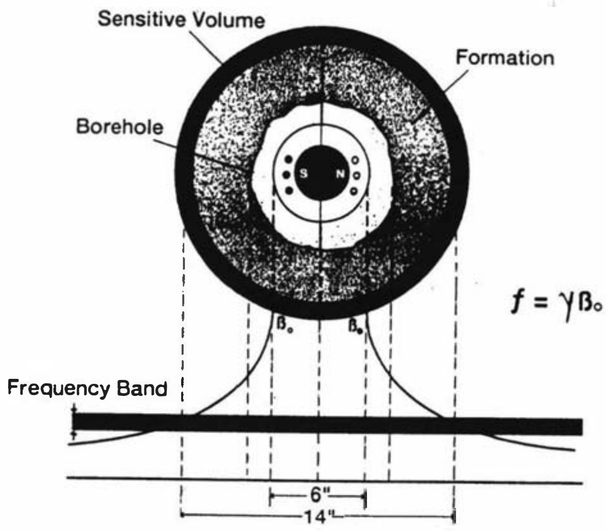


Figure 3.7 Birds eye view of the MRIL tool within a borehole. The sensitive volume is a thin cylindrical shell with a frequency proportional to the distance away from the magnet axis. The typical operating frequency is between 0.5 and 1 MHz. Figure taken from reference 39.

In 1985 a group from Schlumberger [32, 33, 40] starting developing a new well logging tool. Again like the NUMAR group, they were originally inspired by the Los Alamos tool. The first iteration of the design is shown in figure 3.8a where the two cylindrical magnets used in the Los Alamos design are tilted and spaced to produce a resonant volume with greater field strength. The B_1 coil would also be closer to volume resulting in increased coupling. The only drawback is the reduced volume. However this was increased with the next design (figure 3.8b) by using long bar magnets that extended along the borehole to give a volume that also extended along the length of the borehole.

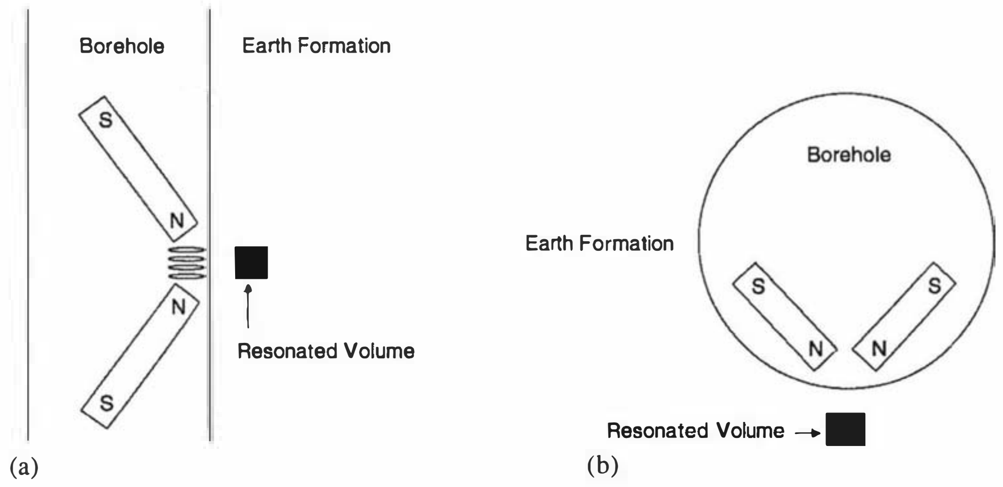


Figure 3.8 Evolution of the Schlumberger well logging tool. (a) Variation to the Los Alamos design, where the magnets are tilted to produce a stronger but smaller sensitive volume. (b) Design using rectangular magnets extending along the borehole. Figures taken from reference 40.

This design has been continually refined to the present "state of the art" form of today, shown in figure 3.9. This probe is called the Combinable Magnetic Resonance (CMR) tool [41]. The tool generates a sensitive zone 28 mm inside the borehole wall and from this region the oil company staff can obtain information such as the porosity of the rock and what types of fluids are trapped within the pores. The field strength of the sensitive region is 55 mT corresponding to a Larmor frequency of 2.3 MHz. The length of the zone is 150 mm. Being 4.8 m long and weighing 204 kg means that it is not really a portable NMR probe, but these types of probes are worth considering as a large amount of money has been invested in the development of these technologies resulting in many ideas that are applicable to the design of smaller probes. These are mainly in the areas of magnet design, experimental techniques and data processing methods. Measurements are performed whilst the tool is being lowered or raised and so a complete log of the hole is obtained. It is a very costly exercise, but in this industry the savings generated justify the cost.

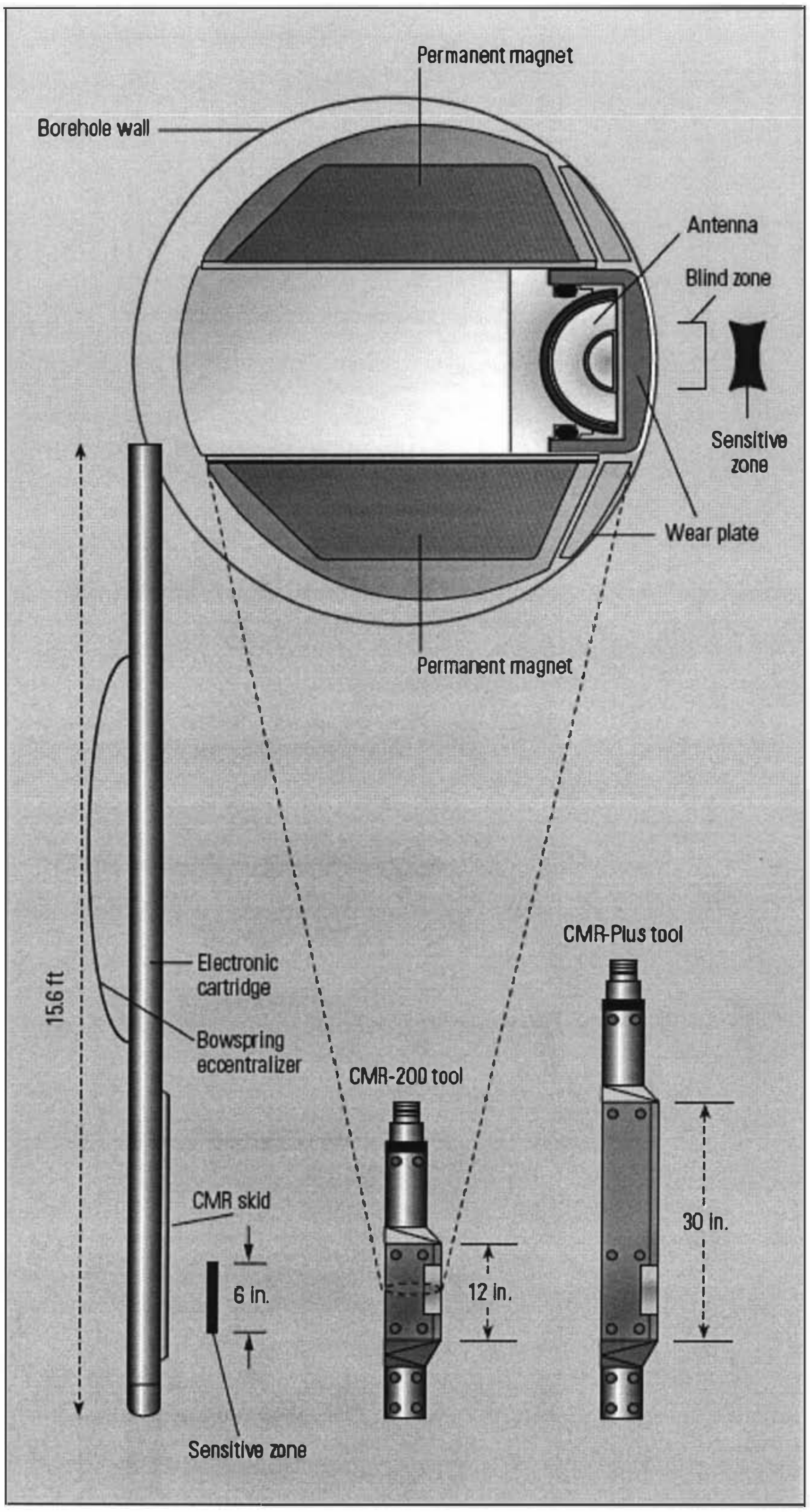


Figure 3.9 The Schlumberger CMR-200 well logging tool. The CMR-Plus tool includes extra magnets to provide prepolarisation of the sample region whilst the probe is raised. Figure taken from reference 41.

3.2.3 The NMR-MOUSE

One example of a truly portable probe is the Mobile Universal Surface Explorer (MOUSE) developed by a group of researchers associated with the Aachen University of Technology [8, 42-50]. It was initially developed for analysing the rubber on car tyres, but has now been applied to areas such as the moisture determination of historic documents and buildings. The device is based on the principles of "inside-out" NMR where the region of interest is external to the probe. The NMR-MOUSE is a surface probe that can only be used to obtain data from samples that are located within a few millimetres of the probe surface. The NMR-MOUSE is relatively easy to construct and is made from readily available materials that cost in the order of a few hundred dollars. Figure 3.10 gives a schematic view of how the probe is constructed and figure 3.11 shows the completed hand held unit. Two rectangular permanent magnets are placed on an iron yoke to generate the necessary B_0 field that is aligned with the surface of the probe. A solenoidal B_1 coil is used to interact with any sample that is placed near the probe surface.

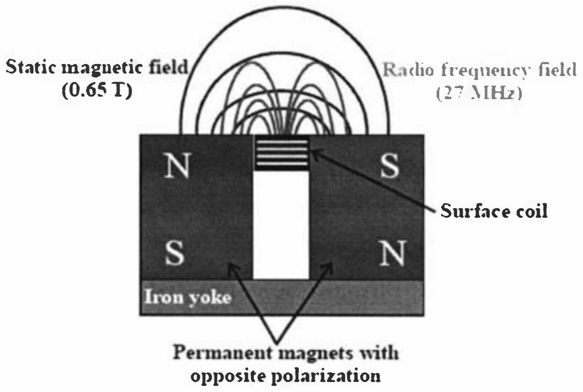


Figure 3.10 NMR-MOUSE, magnet and B_1 coil configuration. Figure taken from reference 49.

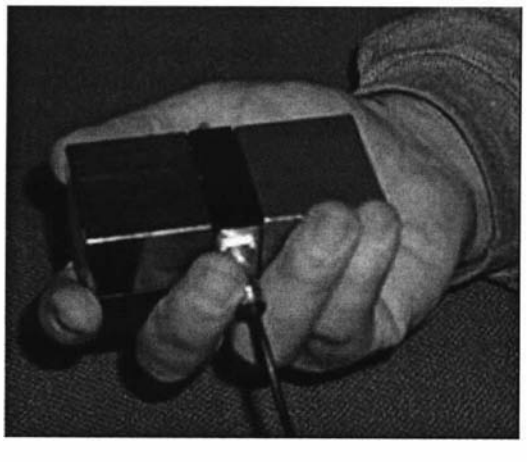


Figure 3.11 Two magnet, handheld NMR-MOUSE. Figure taken from reference 50.

An alternative to this design which again the people in Aachen have produced is the single cylindrical bar magnet mouse shown in figure 3.12. Here a B_1 coil is placed directly on the surface of a magnet and is designed to generate a field that is perpendicular to the B_0 field of the magnet, but this time parallel with the surface of the probe. This is the opposite to the configuration of the earlier, two-magnet MOUSE design. Again only samples within a few millimetres from the surface can be measured. Both NMR-MOUSE types are inhomogenous designs as they have a significant B_0 field gradient within the sensitive volume.

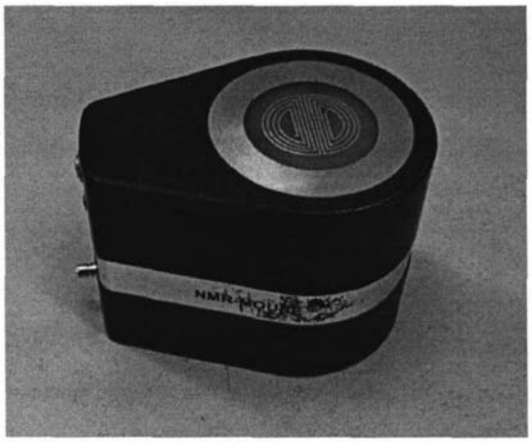


Figure 3.12 The single bar magnet NMR-MOUSE [42]. This probe typically resonates at 19.2 MHz, uses an excitation power level of 100 Watts, has a 90^0 pulse time of 4 μs and produces a Spin Echo signal with amplitude 10 μV and duration 10 μs.

3.2.4 The NMR-MOLE

In the building industry, it is often necessary to determine the moisture content of concrete so that it can be optimally cured for strength, or to know when floor coverings can be applied. Presently the only way to accurately gauge the true moisture content is to use destructive techniques [51]. Typically the industry yardstick of a "month per inch" is used for estimating the time required before the concrete surface can be covered, but it is known that even after a year there can still remain a significant amount of moisture. This is shown in figure 3.13, where profiles were recorded using a relative humidity probe, a destructive technique [51].

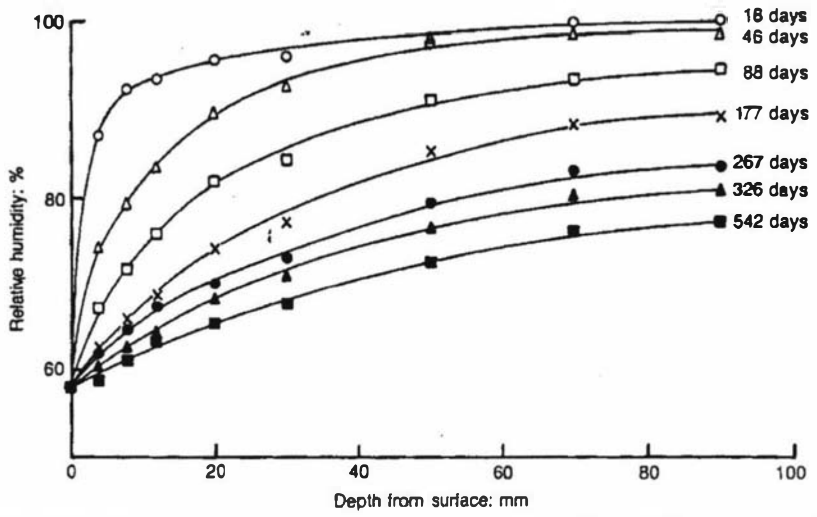


Figure 3.13 The figure shows a series of moisture depth profiles within concrete that were taken at various times after it set. Figure taken from reference 51.

Concrete is inherently porous and so is greatly affected by the environment and the laying process. This makes it very difficult to make predictions based on calibrated standards. Current methods for determining the moisture content are either destructive, require careful calibration or only give an indication of the surface moisture. What is required is a portable instrument that can be taken to a site to non-destructively obtain the moisture content. This would require an NMR probe that could obtain data from deep inside the material. The aim for a new probe, called the NMR MOLE, was to produce a one cubic centimetre region 50 mm into the sample. This depth would be sufficient to enable the monitoring of drying concrete.

The challenge that is facing designers of "inside-out" portable systems is the need to generate a region of homogeneous field remote from the surface of the probe. One approach has been based on the properties of ring magnets [52] and is illustrated in figures 3.14 and 3.15. A ring magnet that has been magnetised along the axis of symmetry will produce a reasonable but small homogeneous region remote from the annular face. The strength and volume of this region can be further improved by adding a cylindrical magnet with the same magnetization direction.

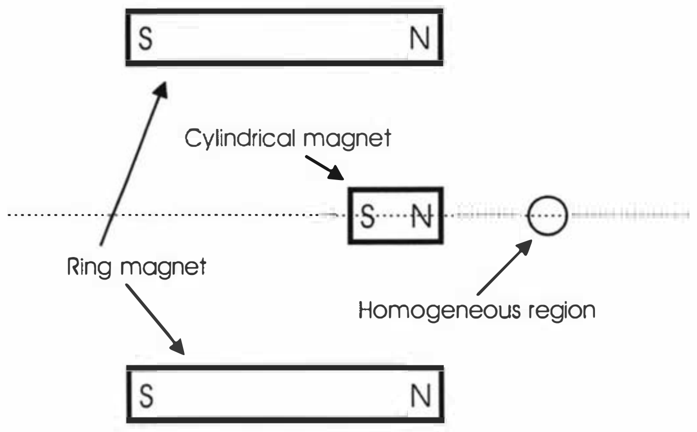
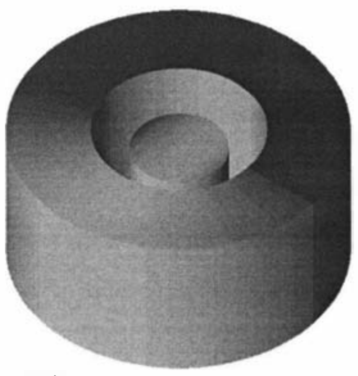
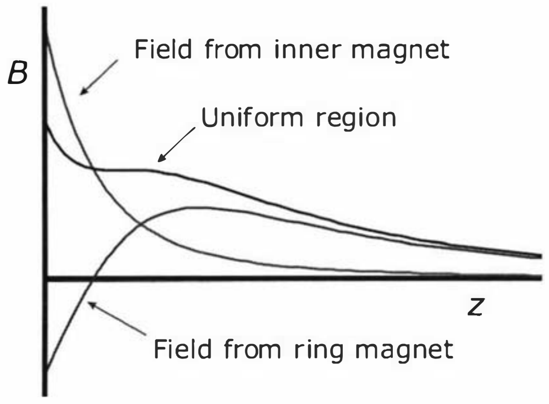


Figure 3.14 Schematic of ring magnet based surface probe. A ring magnet together with a solid cylindrical magnet produces a homogenous region remote from the magnet structure. Figure adapted from reference 52.


(a)



(b)

Figure 3.15 (a) Pictorial view of what the magnet system could look like. (b) Simulated plots of the individual field contributions and the combined field along the magnet axis (z) [53].

A new probe design by Mark Hunter et al [54] uses a series of individual magnets to approximate a ring magnet. A further axial magnet was placed in the centre like the earlier ring magnet design. The position of the central magnet and the axial angle of the ring magnets were adjusted until the desired field was obtained. This is illustrated in figure 3.16. The array was constructed and the design verified using field mapping techniques. A region 55 mm deep, of field strength 10 mT with a volume of approximately 10^{-6} m^3 was achieved. The probe has a diameter of 250 mm and weighs approximately 6 kg.

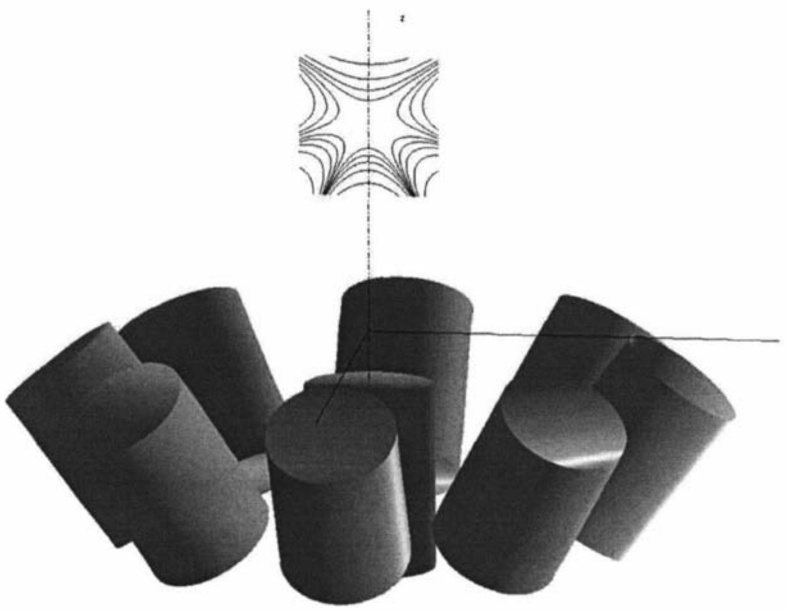


Figure 3.16 Probe magnet arrangement creating homogeneous region 55 mm from the surface [53].

The B_1 coil is not shown, but would be a larger version of the coil used in the single bar magnet NMR-MOUSE. This probe has been "nick-named" the NMR-MOLE as it penetrates deeper into the sample. It also has a larger and more homogenous region than the two types of NMR-MOUSE. The larger sample volume partially compensates for the reduced field strength. Slight variations of the magnet positions have been used to produce a series of different depth probes with correspondingly different field strengths.

3.2.5 Halbach magnet arrays

Another magnet array that is becoming popular is the Halbach array [55]. It is a closed magnet design and therefore has relatively good field homogeneity and strength. It consists of an array of magnets arranged into a ring as shown in figure 3.12 [56]. The magnets themselves are oriented in a fashion so that the fields combine to produce a homogenous region at the centre of the array and in the direction of the arrow shown in the figure. Multiple sets of rings are stacked on top of each other to produce a longer homogeneous field region with typical field strengths of about 0.3 T.

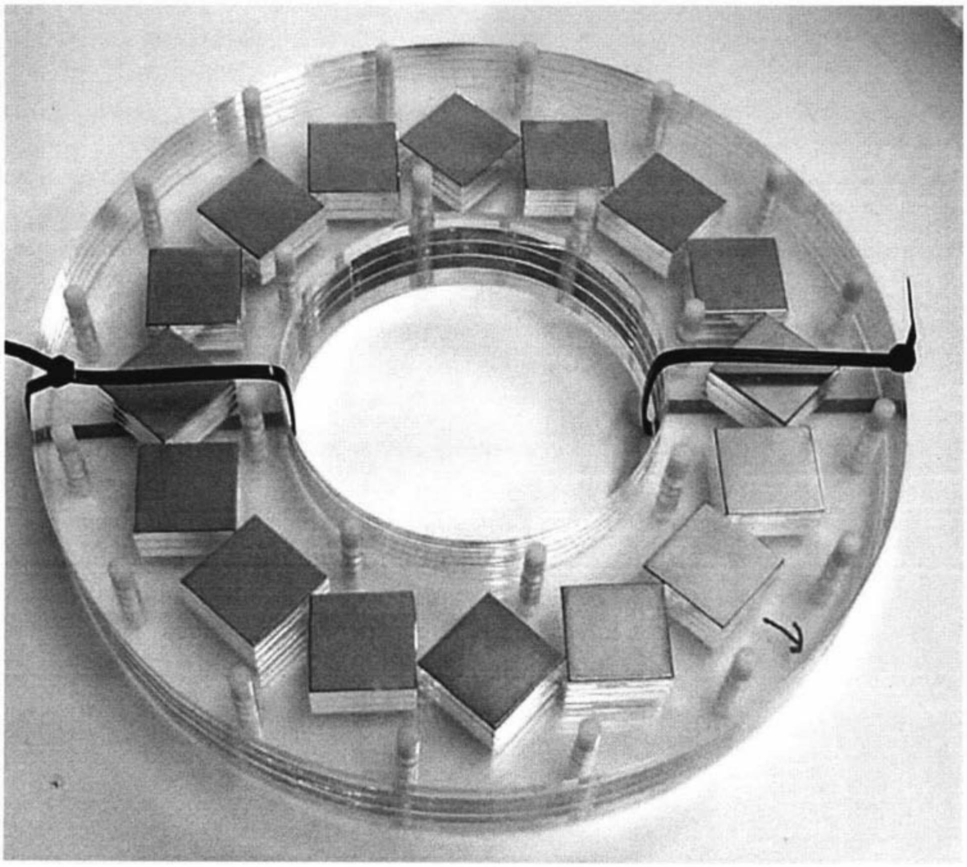


Figure 3.12 Halbach array magnet arrangement [56].

3.3 Outline of a portable NMR system design

Before launching into a new design it wise to clearly define the problem that needs to be solved. In this case the main problem is that there are no compact portable spectrometers on the market that can support the range of permanent magnet probes that are presently being used or developed. There are many difficulties associated in solving this problem. One is that portable NMR is still in its infancy and so there are not many clearly defined portable NMR applications. A lot of work is being undertaken in developing applications. Hence it is a rapidly evolving area, but the actual spectrometer and probe requirements for a lot of these applications are yet to be determined. Usually when building a new probe one would just use one of the commercially available spectrometers of a superconducting system to test it. In the case of the concrete probe (NMR-MOLE) the operating frequency was outside the available range and so it was necessary to first develop a spectrometer before the concrete probe design could be tested. This was not ideal. It meant that the features and specifications of the system had to be determined before the probe was tested. There was a real danger here of producing a system that could do everything one could imagine, but that would have led to a lot of redundancy. Therefore it was necessary to have a flexible design that could easily be adapted to suit different requirements. Once an application is well characterised then a custom solution can be investigated, but, this would depend on the demand for the technology. Being involved with others in all the various parts and phases of developing a total solution is a benefit that some of the companies who are developing spectrometers do not have. The other problem they have is that most of them are used to producing expensive and complicated superconducting based NMR systems and so they have to adjust to a whole new set of design philosophies.

Once the main problem is defined it is then necessary to identify what is required to solve the problem. This involves first looking at what an NMR spectrometer does and then generating a list of design criteria. Every NMR system, whether laboratory based or portable, basically consists of a probe, a whole set of supporting electronics and a user interface for controlling the system and observing/collecting the data. NMR, like many other techniques, requires the stimulation of a sample and then the monitoring of any emissions. It is analogous to the impulse response technique often used to characterise electronic systems. A radio frequency pulse is applied to a coil close to the sample and the subsequent FID signal is received by the same coil and, after amplification, is acquired. For Earth's field NMR this occurs at audio frequencies and so it is relatively easy to construct a spectrometer. Here the signal can be directly sampled using an Analogue to Digital Converter (ADC). Superconducting magnet systems operate in the hundreds of MHz and so the spectrometer is rather complex as it uses many RF stages. High resolution NMR spectroscopy and imaging requires the spectrometer to have good dynamic range and signal to noise performance in order to preserve the signal from the probe. For permanent magnet systems the stimulus/emission is often in the tens of MHz region and so RF transceiver techniques are still required. However the dynamic range and signal to noise requirements are not as demanding as the overall performance is dominated by the relatively poor performance of the probe.

Although these systems are all different and are used for different applications, they all require real time control, data acquisition, signal processing, a data display, a user interface and a probe interface.

After evaluating many existing systems and probes, the main features and specifications that a permanent magnet based system would need were identified. Basically what is needed is a compact, battery-powered, low-cost, light-weight, spectrometer that can be adapted easily for different probes and be easy to manufacture. One of the goals of this work is to make NMR an everyday technology. An ultimate goal would be an NMR based transducer that gave a signal in proportion to a material property with the user being unaware that NMR technology was being used. All the complexity between the application and the user should be able to be hidden. In other words, every effort should be made to hide the complexity from a user who has no interest in the underlying technology.

A further aim of this work was to develop a commercial product and so care was needed to avoid dependencies such as third party software licensing. In summary the main design criteria are:

- Portable
- Compact
- Relatively low cost
- Easy to manufacture
- No hardware/software third party licensing
- Flexible, can adapt for different applications
- Modern interface, be easily interfaced to laptops etc.
- Digital Signal Processing capabilities
- 100KHz to 30MHz frequency range
- Different user levels to hide various levels of complexity.

The problem together with the solution has now been identified, but how should that solution be implemented? The first things is to keep in mind are the special requirements or philosophies of portable NMR. With Portable NMR it is not necessary to have all the features and the superb spectrometer S/N performance offered by superconducting magnet based spectrometer consoles. Portable NMR is also application-specific. Should systems be produced that will hopefully suit all applications, or should systems be made specifically for each application? One extreme results in significant redundancy, while the other extreme results in manufacturing nightmares. A mixture of the two will be most probable. One approach to help with this is to make up building blocks that are chosen and put together. This is done by identifying the common system elements and then making up a series of software and hardware modules. These blocks in themselves are a set of smaller scale problems that need to be solved, each with a set of specifications. Being application-specific makes it important that all the application developers, probe designers and the hardware/software designers keep in close contact to produce optimal solutions.

The second thing that can be done to make portable NMR easier to implement and more affordable is to take advantage of all the new software and hardware technology developments that are occurring. These new developments are often more efficient, cheaper, lower power, more compact, and offer better performance. These developments have resulted in faster computers, broader bandwidth digital communications and compact, lightweight mobile phones. NMR is being driven by advancements in electronics and computing and magnet technologies and so will continue to increase in capability.

A typical layout of an NMR system is shown in figure 3.13 and consists of a number of building blocks. This layout defines a common system architecture that is to be employed for all the portable NMR system designs within this thesis. It basically consists of two main parts, the system core and the transceiver.

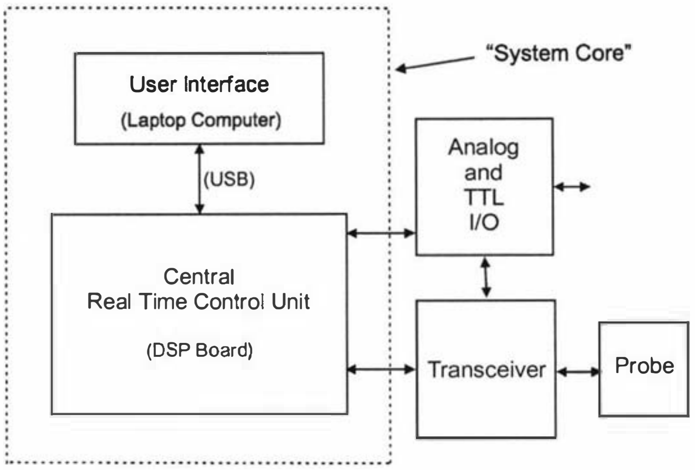


Figure 3.13 NMR system architecture

The generation of precisely timed signals, the capturing of FIDs and the processing/display of data are common to all NMR applications. These functions have been encapsulated within a "system core" module that is described in detail in chapter four [57]. This module goes between the user and the transceiver and can be considered as the digital part of the system. It can also be used for non-NMR applications whenever an instrument controller and user interface is required. The system core takes advantage of the relatively cheap, powerful and flexible capabilities offered by Digital Signal Processor (DSP) technologies to perform high speed real time tasks and laptop computers for data processing and user interfaces.

The transceiver module handles all the signal generation and detection and is more application specific and so often it is here where the differences between the various NMR systems will occur. For Earth's field NMR this part is very simple and consists of an ADC and a DAC (digital to analogue converter) and a handful of operational amplifiers. For the permanent magnet based system a more sophisticated transceiver is required, but some new digital transceiver technologies have become available that will greatly assist in this part of the design. The new digital transceiver technologies have been developed primarily for the mobile phone industry and this is another example of how NMR is benefiting from technology developments occurring in other areas. Using digital receiver technology makes it easier to reconfigure the system in software and provides a great deal of flexibility. For compact systems, the transceiver design is complicated by the increased coupling between the various components. The receiver side has a typical gain of about 90 dB and so great care is needed to minimise unintentional feedback and interference. The probe interface of the transceiver is often referred to as the "RF front end" and is very application specific as the requirements are determined by the probe used.

To complete a design, the enclosure, power supplies and software also need to be developed so that a total solution can be offered. The following four chapters describe the design, construction and testing of a spectrometer developed for permanent magnet based probes. When designing hardware we always need to keep in mind how it is to be manufactured and this introduces further design constraints. Like all engineering design, it is desirable to strive for maximum performance with minimal complexity.

4.0 System core

4.1 Introduction

Real time control and acquisition is more difficult with recent computers since most modern operating systems are unable to provide guaranteed timing of events or direct access to ports and memory spaces. In the past, all one needed to do was to plug in an input-output (I/O) card and write a few lines of code to access it. Today, one must use trusted device drivers and send I/O requests to these drivers from within the application. The operating system then determines when it will perform the request and this results in massive software overheads that significantly reduce the performance. Therefore, in order to meet the timing criteria, it is necessary to implement the critical control/acquisition elements with separate hardware. But the major problem still exists of transferring data between the host computer and the system electronics. With plug in cards, RS232 and the Parallel printer port fast becoming superceded, one is now forced to use one of the high speed serial ports such as USB or FireWire.

Another area that has increased enormously in complexity, but this time to our advantage, is the user interface. There are a number of pre-packaged software environment providers such as Matlab [58] and Labview [59] that assist in the development of these interfaces, but there are disadvantages in this approach. If a product is being developed that is likely to go into production, a significant cost overhead is software licensing. A complete solution from transducer to user has been developed. It consists of a DSP board with a full speed USB1.1 interface to communicate with a host laptop computer, a USB Application Programmers Interface (API) and, if required, a complete user interface development environment and data processing package.

The system core is the controller, data acquisition and processing part of the NMR system. It is responsible for generating all the necessary signals, and for digitising, processing, displaying and storing the captured FID data. In commercial NMR systems it typically consists of a host computer together with some additional electronics.

In order to design a system core that would be flexible and powerful enough to suit a range of NMR applications, a number of people were canvassed for ideas, the details of some existing NMR systems were studied and a list of system requirements was generated. The resulting design criteria are listed below:

- The system core should have a control/acquisition unit that is separate from a host computer. This is to allow different computers to be used as the user interface for the system. Experience with scientific instruments has shown that it is not a good idea to have the control/acquisition unit and the computer all in one, as the life expectancy of the computer is often much shorter than the instrument it controls. Using a separate control/acquisition unit allows the host computer to be upgraded when necessary.
- Ideally the control/acquisition unit should be host platform independent and be able to be used with a wide range of computers and operating systems.

- A standard interface should be used between the control/acquisition unit and the host computer in order to meet the previous criteria. This interface should also be fast, reliable and not involve complicated cabling.
- The system core should be as flexible as possible, so that it can be used as the basis for many different NMR and even non-NMR systems.
- The control/acquisition unit should have a range of digital I/O interfaces such as, for example, a flexible high speed data I/O interface capable of being used with a wide range of analogue-to-digital (A-D)/digital-to-analogue (D-A) units and able to handle a high throughput of data, and some high speed digital outputs to be used in conjunction with the pulse program to control some other units within the NMR system.
- Good dynamic range is necessary for the A-D conversion stage and the signal processing stages that follow it.
- The acquisition part should be able to take advantage of signal processing techniques such as signal averaging, over-sampling, under-sampling and digital filtering.
- The control part should be able to generate a sequence of pulses and waveforms on multiple digital and analogue channels with a timing resolution of 10 ns.
- The system user should have the maximum flexibility when designing NMR experiments and pulse sequences. Looping and nested looping and the ability to rapidly alter parameters between individual experiments through the use of parameter tables, such as delay tables, are essential if multi-dimensional NMR experiments are to be performed.
- A single software package running on the host computer should be able to control the system and analyse the data.
- Rapid updates on the computer screen of acquired data should be possible.
- The system should be compact so that it can be portable.

A decision was made right at the beginning to use a digital signal processor (DSP) as the central element in the control/acquisition unit. The reasons for this are as follows:
- DSPs offer a large amount of processing power at low cost. DSPs are available in packages that are relatively easy to mount.
- DSPs are designed for real time processing of signals and so are ideal for digital filtering.
- 100 MHz processors are available that will give the required timing resolution of 10 ns.
- DSPs, being programmable devices, are very flexible and should easily handle complicated multidimensional pulse programs.
- A range of intelligent high speed parallel and serial interfaces are also provided with most DSPs, so that they can easily be connected to other devices such as ADCs and DACs with minimal "glue logic".

By using a DSP, many functional blocks can be implemented in software instead of hardware. This provides greater flexibility in the design. Two examples of this are the real time digital filtering of the NMR FID and the sinusoidal waveform generation using a digital oscillator algorithm. A 24-bit, 100 MHz, Motorola DSP (type MC56309) was chosen as it has a timing resolution of 10 ns, has good dynamic range, is available in a *quad-flat-pack* surface-mount package and is well supported with a whole host of free development tools such as a C compiler, assembler, simulator and hardware debugger. The MC56309 [62] has 20K x 24-bit words of on-chip program (P) memory and two 7K x 24-bit words data memories (X and Y).

For many applications it is desirable for an NMR system to be portable and battery powered. Obviously a laptop computer would be the most suitable as a system host and so the core of the design was based on a laptop computer and a DSP based control/acquisition unit.

The next phase was to determine what type of interface to use as the communications link between the two units. A number of different interfaces exist, some standard such as Ethernet, and available on many different computer platforms, and some platform specific such as the PC parallel printer port. When choosing an interface, many things need to be considered, such as the throughput, the hardware/software complexity at either end, and the physical cabling required. In the end, there are many tradeoffs between such things as complexity/performance/cost, hardware/software and standard/non-standard/developer support.

For many years most digital I/O devices were easy to program. One would simply write directly to the I/O hardware using in or out (Peek and Poke) commands or memory reads or writes. This was simple, fast, and efficient and required a minimal learning curve. Today with modern operating systems, direct reading and writing to hardware is forbidden and now the only way to communicate with hardware is through the use of device drivers. A device driver is effectively a trusted operating system extension that has more privileges than an ordinary user mode application. Its task is to manage the device and to provide a software interface to users' applications. Device management is important with multi-user, multi-tasking operating systems, as the device may be shared among multiple applications and/or threads. Device drivers were also introduced to prevent applications from the unauthorized use and/or misuse of devices that could potentially bring the whole system down. Through the use of device drivers, operating systems should become more robust and stable. However a big disadvantage of the device driver approach is the huge overhead it places on the system resulting in a much reduced performance of the device and the application. The choice is between stable but slow, or fast and dangerous. It looks like the trend is for more conservative systems where reliability is the main goal. Therefore device drivers are something that we have to learn to live with. One of the unfortunate things about using device drivers to access hardware is that you do not know exactly what or when things may happen. You may send a request to the driver, but it will decide when it will do it, depending on the other tasks that the system has to perform. This means that guaranteed timing and precise synchronisation of events is impossible. Taking all this into account, any hardware interface that is used will need to be well supported by a good device driver that is optimised for good data transfer rates and has a straightforward application programmer's interface (API).

A number of different interfaces were evaluated in order to choose the one that best met the design criteria. These are listed as follows:

- PC parallel printer port. Unfortunately this is limited to PCs only and will soon be obsolete. Manufacturers want to move away from using this port, as it takes up a lot of board and panel real-estate, and requires a large expensive cable. The support for Windows98/2000/XP is not very good, as there are no readily available general purpose device drivers.

- RS232. This interface is too slow and is also becoming obsolete, as it is no longer supported by some platforms, such as new computers from Apple Computer [60] and more recently most PC manufacturers.
- Ethernet. This is a possible solution, as it is very common and available on most computer platforms. But there are a number of big drawbacks, one of which is that the bandwidth can not be guaranteed if the system is connected to a network. In addition it requires a large amount of complicated software at both ends to support it, as there are many protocol layers to work through and hence massive overheads.
- USB1.1. This is a common interface that has been adopted by most platform manufacturers and therefore will be around for a while. USB1.1 is a 4 wire serial interface that is inexpensive to implement, compact and well supported by Windows98/2000/XP, MacOS and Linux. Reasonable data transfer rates of 12 Mbits/s can be obtained. USB peripheral chips are available that can be easily interfaced to DSP devices and there is also a whole host of developer information available.
- FireWire (IEEE 1394). This is a high speed serial interface that was mainly developed by Texas Instruments together with Apple Computer. Unfortunately it is not supported by all PC providers as it is a rival technology to USB which is strongly supported by Intel. FireWire also has the disadvantage that the software/hardware requirements for the device end are a lot more demanding than USB1.1. However it does have much better transfer rates than USB1.1 and so is still a reasonably attractive option if high transfer rates are required.

USB1.1 was chosen as the interface between the laptop and the DSP based control/acquisition unit, as it looked to be the most suitable. More recently USB2.0 has become available and it has the capability to transfer data at the rate of 480 Mbits/s. It was not available at the time of developing the system core therefore it was not considered. Shortly wireless USB will be available and this could be the ultimate interface for the future.

One of the design criteria was that the control/acquisition unit should be able to be interfaced to a range of ADCs and DACs so that the system could easily be adapted to a range of NMR applications. The idea therefore was to make a DSP board with a USB interface, and also to bring most of the DSP I/O lines out to a connector, so that it could be connected to a separate ADC/DAC board. This way only the ADC/DAC board would ever need to be changed. Using this design would provide a fairly general purpose signal processing and control platform that could be used for many applications other than NMR such as, for example, the controller for a scanning tunnelling microscope [57].

The additional main components and I/O ports of the board are (figure 4.0):

- 256K x 24-bit high speed memory that can be software configured as extra X,Y or P memory.
- 2 synchronous serial ports that can be used with many ADCs/DACs.
- A hardware debug port.
- 256K x 8-bit flash memory that is used to store the operating software which is loaded into the processor upon Reset.
- A full speed USB1.1 interface using the Philips USB D12 device.
- A high speed I/O port consisting of buffered address, control and data lines.

The board was designed as a standard eurocard sized four-layer circuit board with a DIN41612 connector at one end for all the I/O lines and by combining with other plug in modules a complete system can be easily built up in a standard rack.

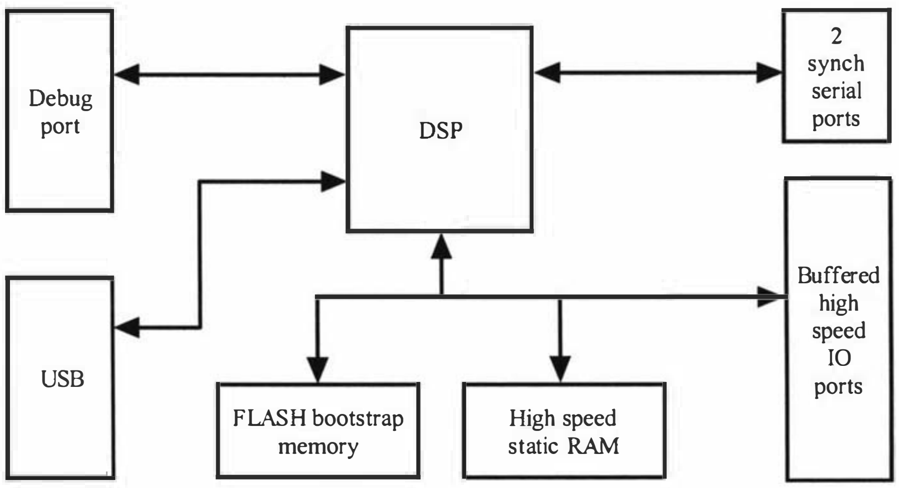


Figure 4.0 DSP board block diagram.

For NMR, the pulse programs would be executable program code written for the DSP using an assembler or C compiler on the laptop. The code would then be downloaded to the DSP board when required. For the user interface and controlling application on the laptop a modified version of Prospa was used. (Prospa is an NMR processing application written in-house. It is now available from magritek). Using Prospa saved a considerable amount of time as the only code that had to be written was that dealing with the DSP board, and not a whole visual user interface.

4.2 System core hardware

4.2.1 Motorola DSP56309 DSP and evaluation Board

The DSP56309

The DSP56309 is one of a series of 24-bit DSP chips from Motorola (now Freescale [61]) that can execute most instructions in a single clock cycle. As the intention was to build a custom DSP board, the chip packaging is a very important consideration. Most high end devices are now only available in Ball Grid Array (BGA) packages, which are extremely difficult to work with as they require special techniques and equipment to solder them to a circuit board. However the 56309 is available in a 100 MHz, 144 pin quad-flat-pack version and this makes it one of the fastest DSPs available in a standard surface mount package. It can be soldered to a board without any special equipment apart from some soldering paste and a reflow soldering tip. A DSP chip was also desired with a data width greater than 16 bits. Floating point processors were avoided as they are not as efficient at performing fixed point operations as a fixed point processor, and often have much poorer response times to interrupts which can make high speed real time applications more difficult. Floating point processors are also physically large devices with many pins and consume much more power than a fixed point device. A 24-bit processor was ideal as it had the dynamic range needed for signal averaging, while for digital filtering 24-bit coefficients could be used resulting in better filter performance. The 56309 is similar to the older 56000 series devices with which I have some experience, so this was another consideration when choosing this device. Another point in its favour is the availability of good, free development software such as a C compiler, an assembler and a debugger, as well as lots of application notes and software examples of signal processing algorithms. Overall the DSP56309 performs well and is well suited to NMR signal processing and control applications. It is also well supported by a large international company and many semiconductor retailers.

The DSP56309 [62, 63] consists of a DSP56300 core and a peripheral and memory expansion area as shown in figure 4.1. The 56309 is one of the 56300 family and so shares a common core but has different peripherals and memory from other devices. The memory is divided up into three parts: program, X data and Y data. This is common with DSP devices as many operations, such as an instruction fetch and an X and Y data move, can be done in parallel. It is this level of parallelism that makes DSP devices so fast. The separate data memories are very useful when implementing digital filters as one memory may contain the filter coefficients and the other the data. Part of the program memory can be used as an instruction cache and it is also possible to make the X and Y data memories larger by allocating some of the program memory to them. Off-chip memory expansion is also possible through the use of the external data, address and control buses. The peripheral expansion area consists of four units. The first is the triple timer module which consists of three independent 24-bit timer/event counters each with an associated I/O pin. The timers can perform a number of functions such as counting, pulse width measurements and pulse width modulation. The next module is the host interface which consists of an 8-bit I/O port and a number of control lines. The host port has been designed to allow a glue-less connection to a number of industry standard microcomputers, microprocessors and DSPs that may operate as the master (or host) processor. In this configuration the DSP behaves as a memory addressed peripheral to another processor. The host port can also be configured as 16 general purpose I/O lines and it is this mode that I have used to interface the DSP to the USB chip. The next module is the dual enhanced synchronous serial interface which is often used to communicate with high speed serial devices such as ADCs and DACs. The last module is an asynchronous serial communications interface that can be used as an RS232 port. The external address, control and data buses are common to all the devices within the 56300 family, as well as the interrupt pins, the Joint Action Test Group (JTAG) port, and the On Chip Emulation module (OnCE).

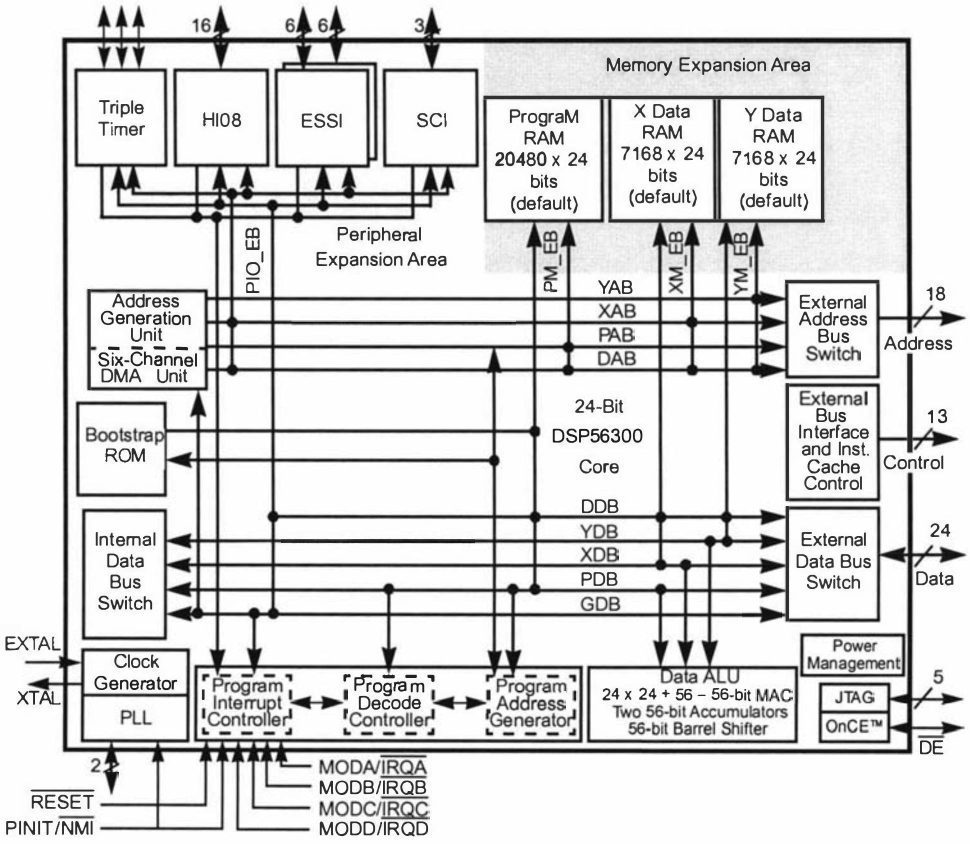


Figure 4.1 The DSP56309 block diagram. Figure taken from reference 62.

The JTAG port can be used to verify the device's hardware and is a feature that is mainly used during manufacturing. The OnCE port can be used for debugging as it allows one to examine and change registers, memory, and on chip peripherals and as such is a very useful feature.

The DSP56309 programming model (figure 4.2) is the same as all the other devices in the 56300 family [64] and therefore all the software is code compatible. This means that all the software written for 56309 can easily be ported to planned future higher performance devices.

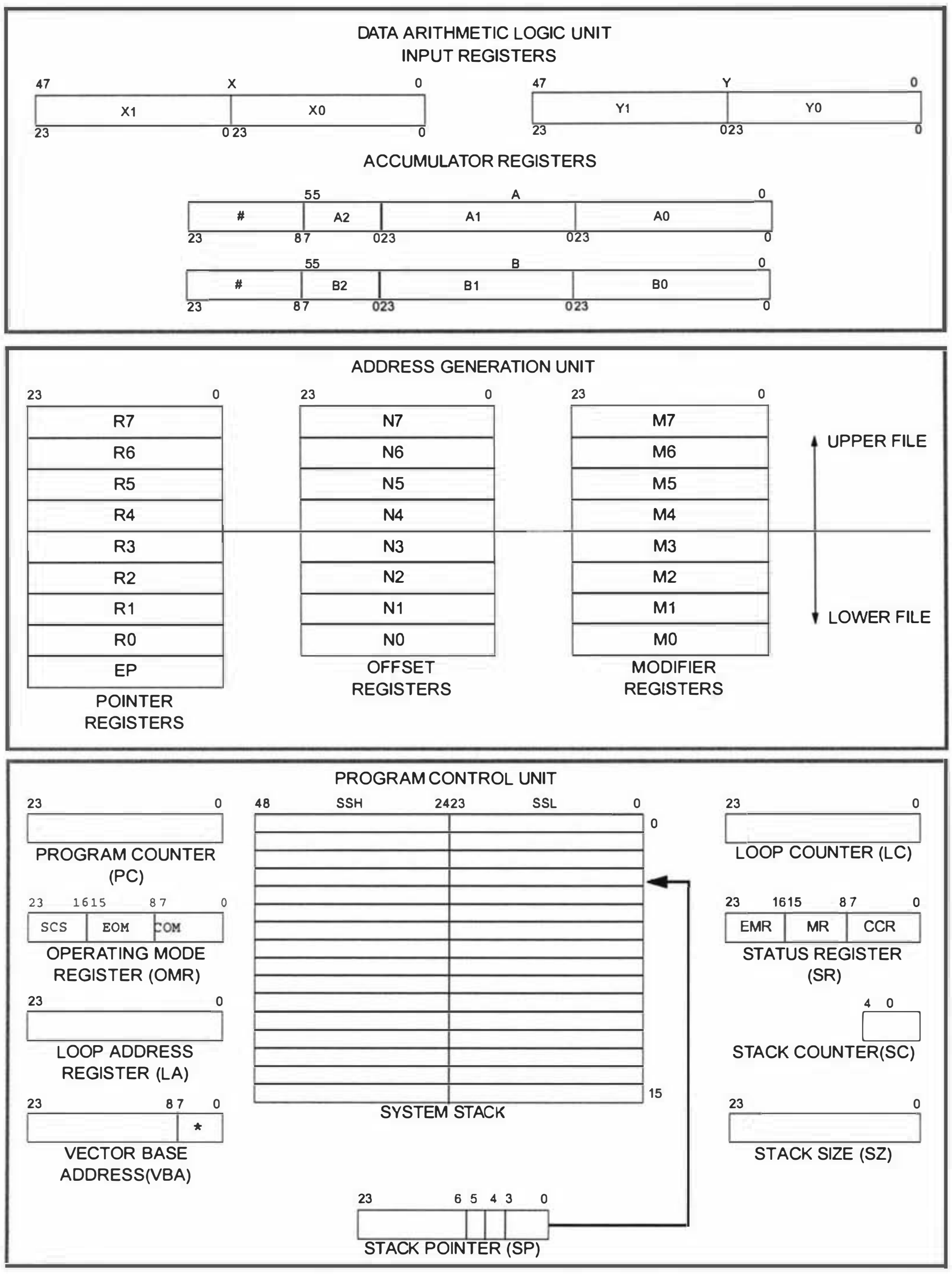


Figure 4.2 The 56300 series programming model. Figure taken from reference 64.

The programming model is divided into three sections, each associated with a different device unit. The Data Arithmetic Logic Unit performs all the arithmetic and logical operations on the data and is made up of a number of 24-bit registers and two 56-bit accumulators. One of the most popular operations that has been optimised for the DSP is the multiply-accumulate (MAC). This is where the contents of two 24-bit input registers are multiplied and then added to the contents of one of the accumulators. During the multiplication phase two new values can be read from the X and Y data memories and loaded into the input registers, ready for the next multiply. If the MAC operation is performed repeatedly (in a loop) it is then possible to perform one multiply accumulate per clock cycle. This technique has become the core of digital filtering where one needs to quickly multiply and accumulate an array of filter coefficients with an array of signal data. The next section is the address generation unit, which consists of a number of 24-bit registers that are used as memory pointers. Offset and modifier registers are also included and are used when implementing circular buffers and/or modulo-n indexing. Circular buffers are very useful for digital filtering, as a first-in first-out (FIFO) buffer is needed for the signal data. The last section is the program control unit, which is very similar to that of a standard microprocessor with its program counter, stack pointer and status register. The loop counter and loop address registers are used to implement zero overhead hardware "DO n" looping. Again this is another very useful feature for digital filtering.

The evaluation board

A DSP evaluation board was purchased to become familiar with the device and programming environment. Unfortunately only an evaluation system for the DSP56303 could be obtained. This device is functionally identical to the DSP56309 except that it operates at 80 MHz and has less on-chip memory. The Motorola DSP56303 evaluation board [65] was designed for real time audio signal processing applications and has the following valuable features:

- An on-board stereo 16-bit ADC/DAC capable of 48K samples per second on each channel simultaneously.
- It uses the Motorola 24-bit DSP56303 chip [66].
- Most of the DSP chip interfaces are brought out onto connectors, allowing other units to be connected to the evaluation board.
- It has 32K words of high speed static RAM, ideal for storing FIDs.
- It comes with a whole host of development software.
- It is inexpensive.

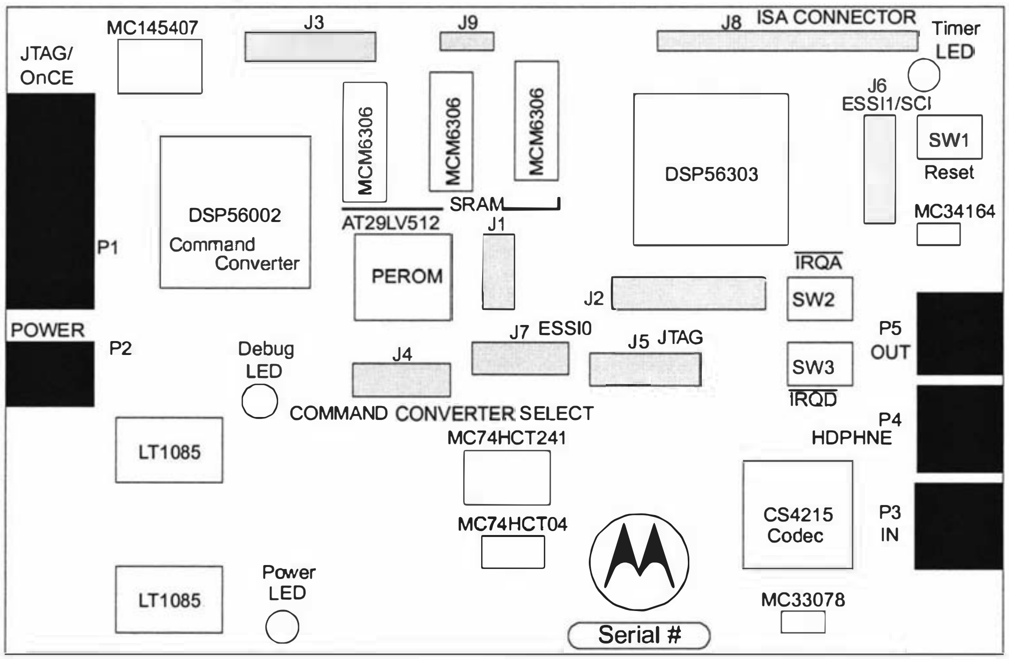


Figure 4.3 Motorola DSP56303 evaluation board layout. Figure taken from reference 65.

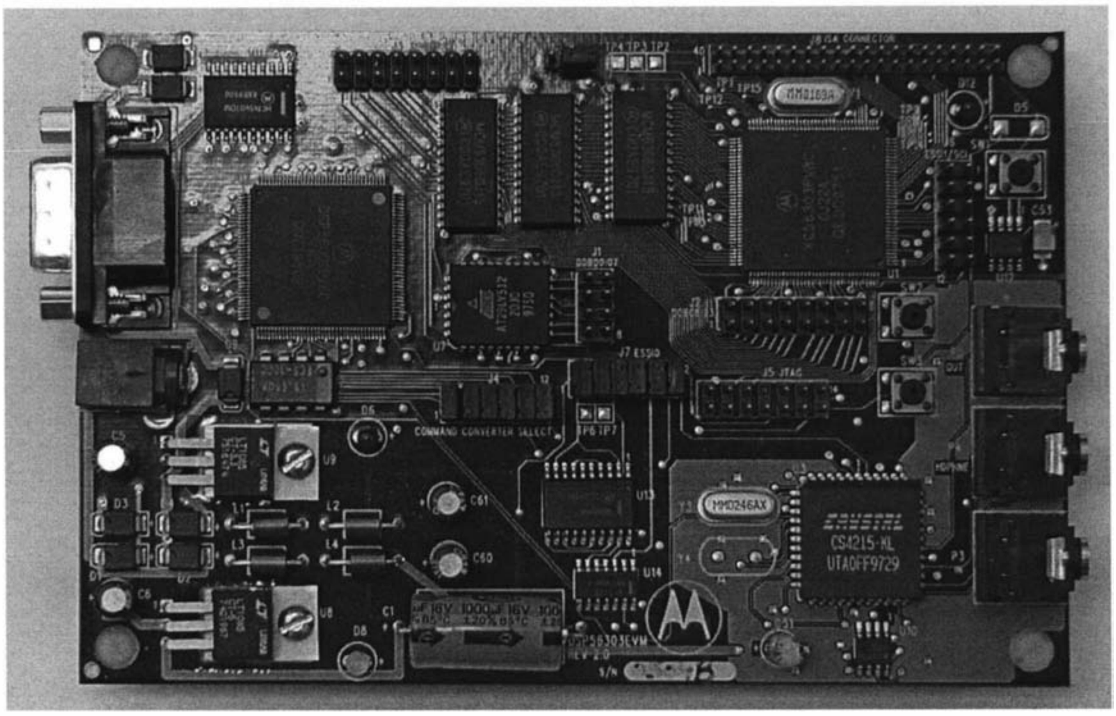


Figure 4.4 Motorola DSP56303 evaluation board [65].

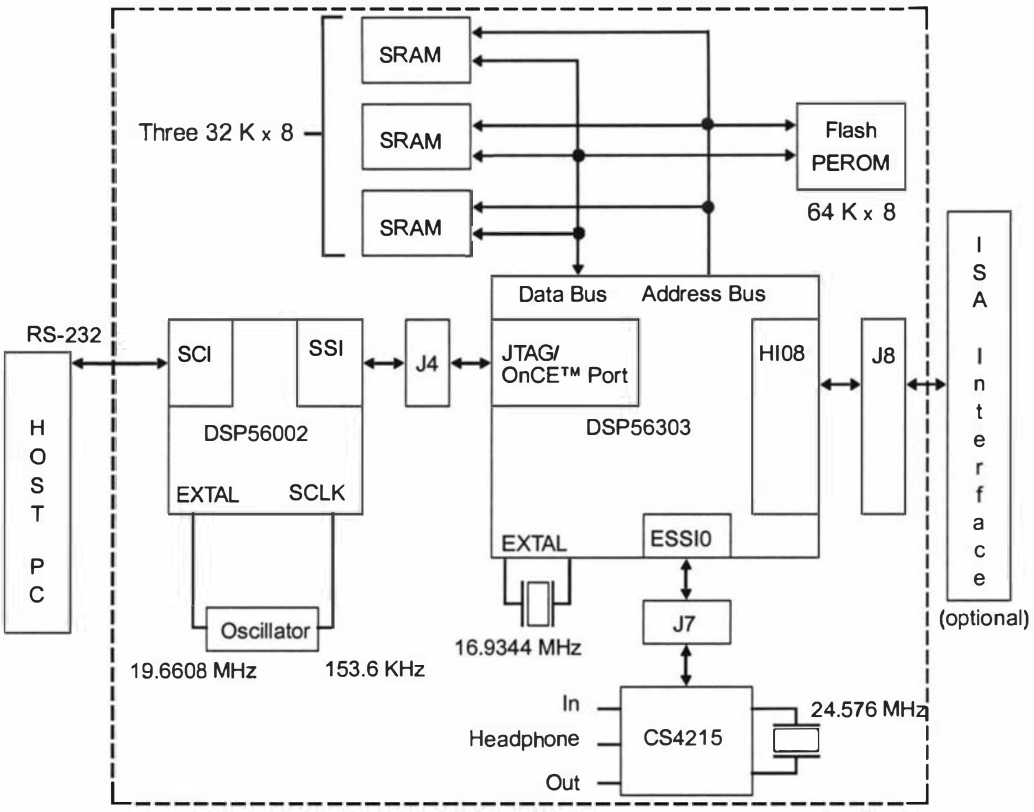


Figure 4.5 Motorola DSP56303 evaluation board simplified schematic. Figure taken from reference 65.

The DSP56303 evaluation board as shown in figures 4.3 to 4.5 is an evaluation system produced by Motorola and is designed to help developers to become familiar with the DSP device hardware and software and thus kick start the development of custom DSP boards. In addition to a DSP56303 chip it includes some other peripherals such as some high speed SRAM, a bootstrap EEPROM, a JTAG/OnCE command converter and a stereo audio ADC/DAC converter chip. The host port, the reset and three interrupt input pins are brought out onto a two row 2 mm pitch 40-way male header connector (J8). The JTAG/OnCE command converter is an older DSP device that is used to interpret commands sent from a host PC via an RS232 link, and then control the 56303 chip via the JTAG/OnCE port. This is used extensively when debugging, as the debugger allows the user to down-load programs into the 56303's memory and then execute them using breakpoints and single stepping. The 56303's registers and memory can also be monitored and modified using the debugger. The EEPROM is a "bootstrap" ROM that contains program code that can be loaded into the 56303's memory and then executed following a reset. The CS4215 stereo ADC/DAC is described by its manufacturer [67] as a 16-bit multimedia audio codec that was developed for high quality music processing. The sampling rate is programmable and ranges from 8 to 48 kHz. Each of the two ADC channels consist of a programmable gain stage followed by a 64 × over-sampled sigma-delta converter. The DAC channels each consist of a delta-sigma converter followed by a programmable attenuator. The CS4215 communicates with the 56303 via a synchronous serial interface. The default memory map of the 56303 evaluation board is shown in figure 4.6, where the external 32 K words of static RAM appears as a unified block of X and Y data memory between addresses $10000 and $18000. This however can be changed by programming one of the address attribute registers (AAR0) of the 56303.The amount of internal program and data memory can also be changed by suitably programming the cache enable and memory switch bits of the 56303's control registers. The external static RAM is very useful for storing acquired FIDs, as there is not enough memory within the DSP to do this.

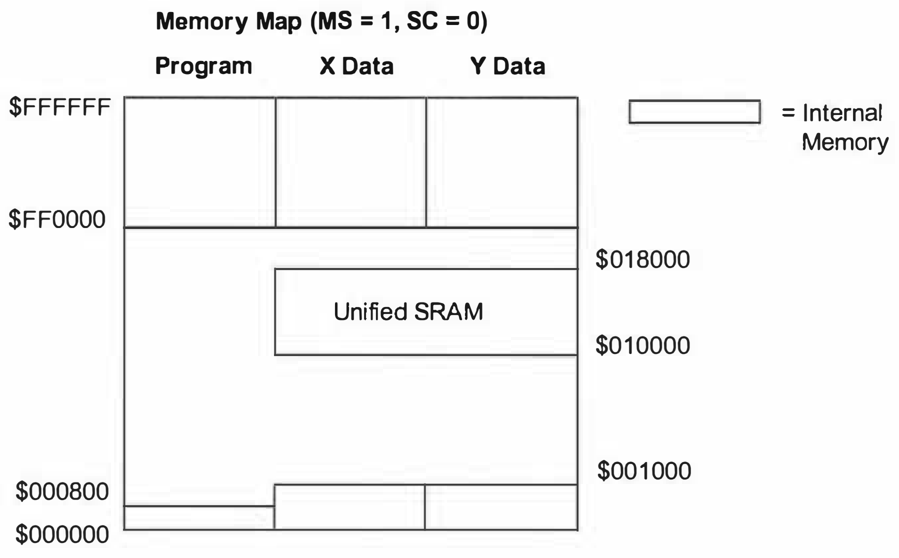
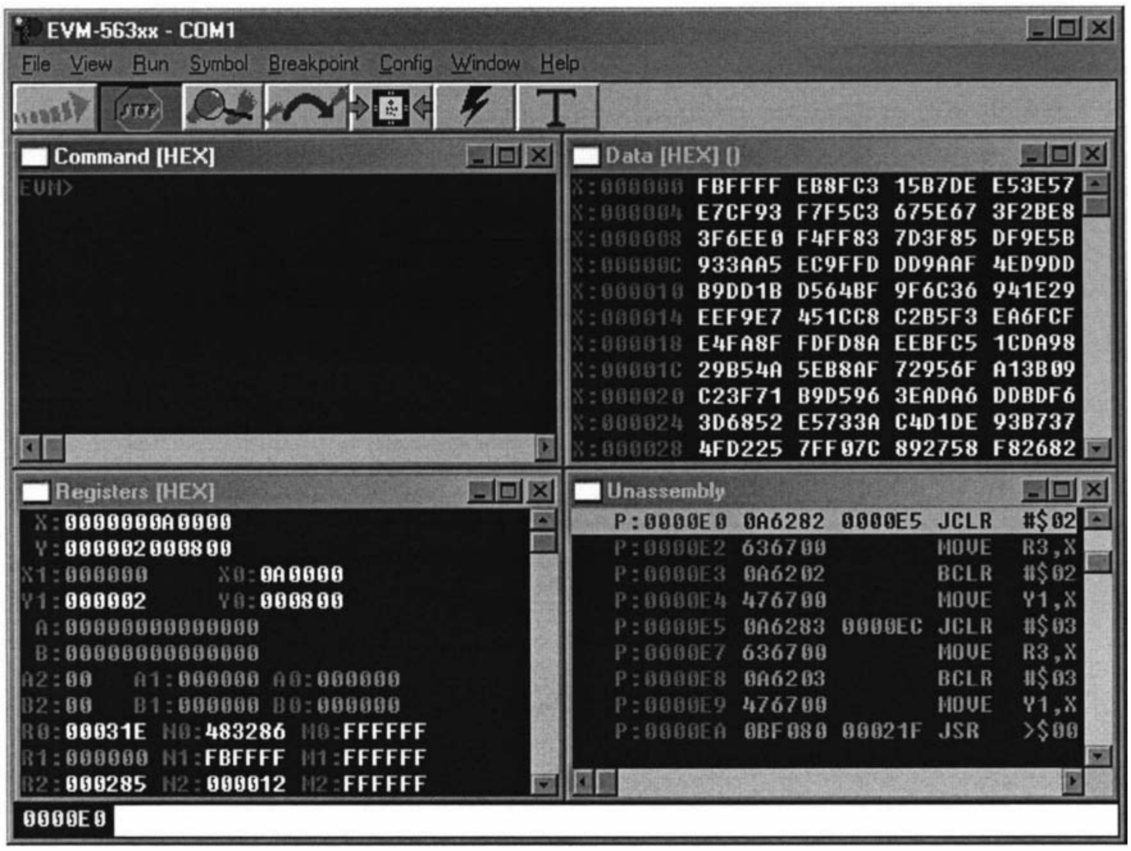


Figure 4.6 Motorola DSP56303 evaluation board memory map. Figure taken from reference 65.

The debugger software uses a graphical interface to monitor and control the operation of the DSP device within the evaluation board. A typical window is shown in figure 4.7 and shows how one can look at the contents of the internal registers and memories. Some other useful features of the debugger are its ability to single step through code, to set breakpoints and its in-built assembler/disassembler.



```
EVM>
```

```
X:000000 FBFFFF EB8FC3 15B7DE E53E57
X:000004 E7CF93 F7F5C3 675E67 3F2BE8
X:000008 3F6EE0 F4FF83 7D3F85 DF9E5B
X:00000C 933AA5 EC9FFD DD9AAF 4ED9DD
X:000010 B9DD1B D564BF 9F6C36 941E29
X:000014 EEF9E7 451CC8 C2B5F3 EA6FCF
X:000018 E4FA8F FDFD8A EEBFC5 1CDA98
X:00001C 29B54A 5EB8AF 72956F A13B09
X:000020 C23F71 B9D596 3EADA6 DDBDF6
X:000024 3D6852 E5733A C4D1DE 93B737
X:000028 4FD225 7FF07C 892758 F82682
```

```
 X:0000000A0000
 Y:000002000800
X1:000000      X0:0A0000
Y1:000002      Y0:000800
 A:00000000000000
 B:00000000000000
A2:00    A1:000000 A0:000000
B2:00    B1:000000 B0:000000
R0:00031E N0:483286 M0:FFFFFF
R1:000000 N1:FBFFFF M1:FFFFFF
R2:000285 N2:000012 M2:FFFFFF
```

```
P:0000E0 0A6282 0000E5 JCLR  #$02
P:0000E2 636700        MOVE  R3,X
P:0000E3 0A6202        BCLR  #$02
P:0000E4 476700        MOVE  Y1,X
P:0000E5 0A6283 0000EC JCLR  #$03
P:0000E7 636700        MOVE  R3,X
P:0000E8 0A6203        BCLR  #$03
P:0000E9 476700        MOVE  Y1,X
P:0000EA 0BF080 00021F JSR   >$00
```

```
0000E0
```

Figure 4.7 Motorola DSP56303 evaluation board debugger software interface [65].

Parallel port command converter

As a custom DSP board was being developed, a way was needed to initially program the FLASH ROM on the board and to debug problems. A parallel port command converter was purchased. This is an interface module (figure 4.8) that goes between the parallel port of a host computer and the JTAG interface of the DSP device on the board.

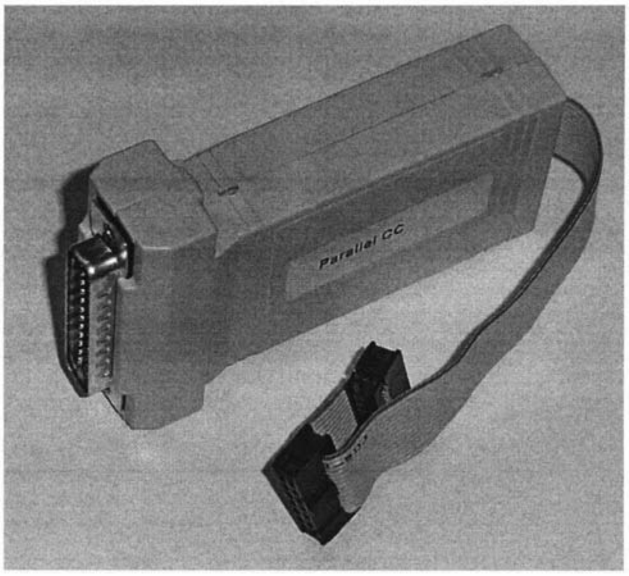
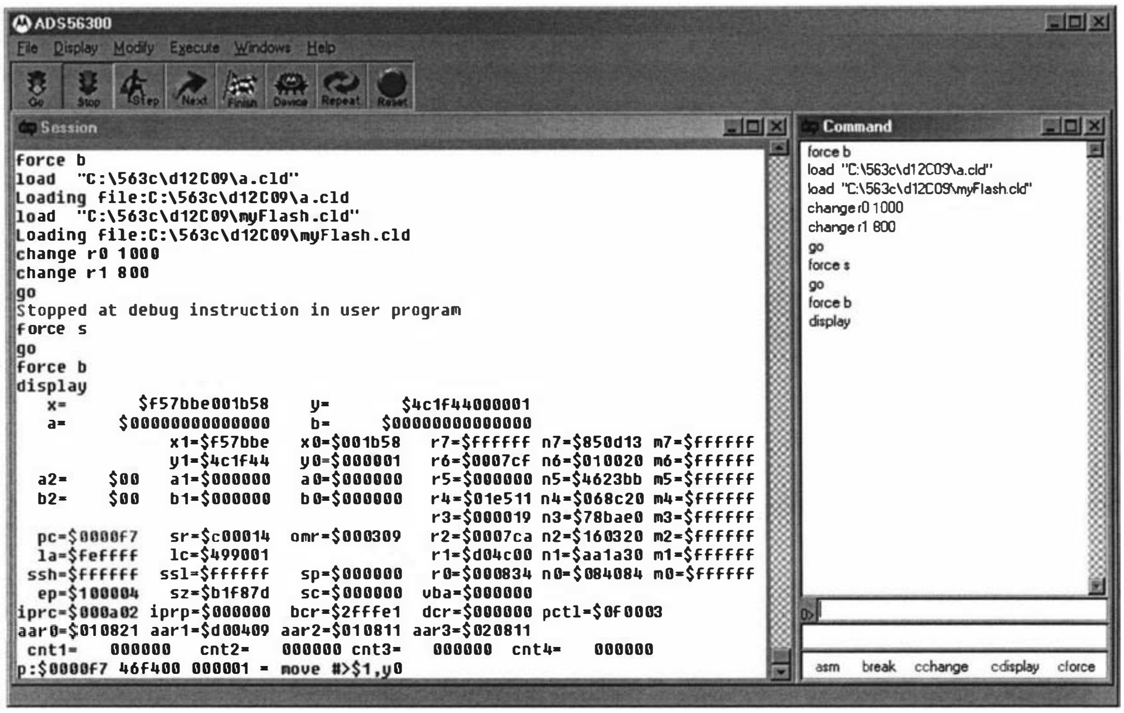


Figure 4.8 Parallel port command converter module [68].

The parallel port command converter module is nothing more than a few logic buffers and some open collector transistors and is therefore easy to reproduce. The software developers suite that came with the DSP56303 development system also included debugging software that worked with the parallel port command converter. This debugger was very similar to the previous one used, but was more advanced and refined. A typical screen window is shown in figure 4.9 where the operating system has been programmed into the board's memory so that after a RESET it would automatically run the DSP/USB operating system.


ADS56300
File Display Modify Execute Windows Help
Go Stop Step Next Finish Device Repeat Reset
Session
Command
asm break cchange cdisplay cforce

```
force b
load  "C:\563c\d12C09\a.cld"
Loading file:C:\563c\d12C09\a.cld
load  "C:\563c\d12C09\myFlash.cld"
Loading file:C:\563c\d12C09\myFlash.cld
change r0 1000
change r1 800
go
Stopped at debug instruction in user program
force s
go
force b
display
   x=         $f57bbe001b58    y=         $4c1f44000001
   a=     $0000000000000000    b=     $0000000000000000
                x1=$f57bbe   x0=$001b58   r7=$ffffff n7=$850d13 m7=$ffffff
                y1=$4c1f44   y0=$000001   r6=$0007cf n6=$010020 m6=$ffffff
  a2=     $00   a1=$000000   a0=$000000   r5=$000000 n5=$4623bb m5=$ffffff
  b2=     $00   b1=$000000   b0=$000000   r4=$01e511 n4=$068c20 m4=$ffffff
                                          r3=$000019 n3=$78bae0 m3=$ffffff
  pc=$0000f7    sr=$c00014  omr=$000309   r2=$0007ca n2=$160320 m2=$ffffff
  la=$feffff    lc=$499001                r1=$d04c00 n1=$aa1a30 m1=$ffffff
 ssh=$ffffff   ssl=$ffffff   sp=$000000   r0=$000834 n0=$084084 m0=$ffffff
  ep=$100004    sz=$b1f87d   sc=$000000  vba=$000000
iprc=$000a02 iprp=$000000  bcr=$2fffe1  dcr=$000000 pctl=$0f0003
aar0=$010821 aar1=$d00409 aar2=$010811 aar3=$020811
 cnt1=   000000   cnt2=   000000 cnt3=   000000  cnt4=   000000
p:$0000f7 46f400 000001 = move #>$1,y0
```

```
force b
load "C:\563c\d12C09\a.cld"
load "C:\563c\d12C09\myFlash.cld"
change r0 1000
change r1 800
go
force s
go
force b
display
0>
```

Fig 4.9 Suite56 debugger [68].

4.2.2 USB1.1

USB is one of the newer interfaces available and most of the documentation is rather poor and lacking in detail and examples. But a few books [69, 70] are available that provide some help and examples. Some web sites [71] are also available with further information and resources for USB developers. Reverse engineering is often required to work out some details of the USB hardware/software protocols.

USB is a network that uses a tiered star topology to connect multiple devices. This is done with the use of hubs, which are the nodes of the stars within the network. Differential signalling is used to transmit and receive packets between a host controller and a device, in a half duplex arrangement that is capable of transferring data at 12 Mbit/s. The packets are organised into 1 ms duration frames and there are different types of packets, which are listed as follows:

- The start-of-frame token packet (SOF) is sent by the root hub every 1 ms and is used as a "heartbeat" signal to synchronise devices. Within the packet is an 11 bit frame number counter that rolls over every 2 seconds and can be used by devices as a real time clock. This packet is very short in total, to allow the possible transmission of 1500 bytes of data between each SOF packet.

- The root hub also sends out other token packets when it wants to communicate with devices. The first two of these are the IN and OUT token packets, and are used by the controller to indicate that it wants to read or write data from/to the device. The last of these is the SETUP token packet, which is used to send data (usually a request) to the device, which it must accept and action.
- The next type of packet are the DATA0 and DATA1 packets. They are used by the controller or device to transfer data and they alternate to provide a way of detecting lost packets. The length of the data packet can vary from 0 to 1023 bytes, this is to allow for the variation in required transfer sizes and therefore bandwidth wastage is avoided. Multiple data packets can be sent during a single 1 ms frame.
- The last type of packet are the handshake packets, ACK, NAK and STALL, and are used to indicate successful or various levels of unsuccessful transmission.

Each packet starts with an 8-bit SYNC sequence that is used to synchronise the transmit and receive clocks, this is then followed by an 8-bit packet identifier. Within a 1 ms frame, multiple packets can be sent between the controller and the devices.

USB devices can have a number of different “endpoints”, which are individual buffers that allow parts of the USB device to be uniquely addressed. Using this method multiple transfer types can be in progress simultaneously. Before a data transfer can occur, the host and the device must establish a “pipe”, which is a logical connection between a device’s endpoint and the host controller’s software.

Each USB device contains a set of “descriptors”, which are tables that contain information about the device, its configuration and its endpoints. When a device is connected to a USB network, the host controller interrogates the device and obtains the information stored in the descriptors.

USB uses four different types of transfers, which are predefined sequences of packets used to define data movement between the controller and a device’s endpoint. These are listed as follows:

1. Interrupt transfers. This is when the controller is set up to poll the device periodically (up to 1000 times a second) to determine if it needs attention. This would be used for devices such as computer mice, where only small amounts of data need to be transferred but without any significant delay.
2. Bulk Transfers. This type is used to transfer large amounts of data, but without any guaranteed delivery time or rate. It is a low priority transfer type, which makes use of any available spare bandwidth that is not being used for other transfers. If many busy devices are connected to the USB network, bulk transfers can become very slow. But if network activity is low, transfer rates of greater than 1 Mbyte/s are easily obtainable. This type of transfer is commonly used for devices such as printers and scanners.
3. Isochronous Transfers. This is used to transfer a packet of data between the controller and a device every frame. Here the host computer ensures that there is enough available bandwidth before it agrees to the connection. Once set up the device is guaranteed a slice of every frame. This type of transfer is used for transferring real time data, such as that from modems or audio/video systems.
4. Control Transfers. These are the most complicated transfers and are used for the system control of devices. The controller uses these types of transfers to request devices to perform various standard and vendor defined functions. A control transfer consists of three transaction phases. Firstly a setup phase, which consists of a SETUP packet, a DATA0 packet and a handshake (ACK) packet. The DATA0 packet

(figure 4.10) always contains 8 bytes, which specify the transfer request type, the actual request and some other parameters particular to the request. The next phase is an optional data phase and is used to transfer data in either direction between the controller and the device. The setup phase will specify if the optional data phase is required and the transfer direction. The last phase of the transfer is the status phase, which is used to indicate the successful execution of the requested function and the data transmission. Each USB device has a control endpoint (endpoint0) that handles control transfers.

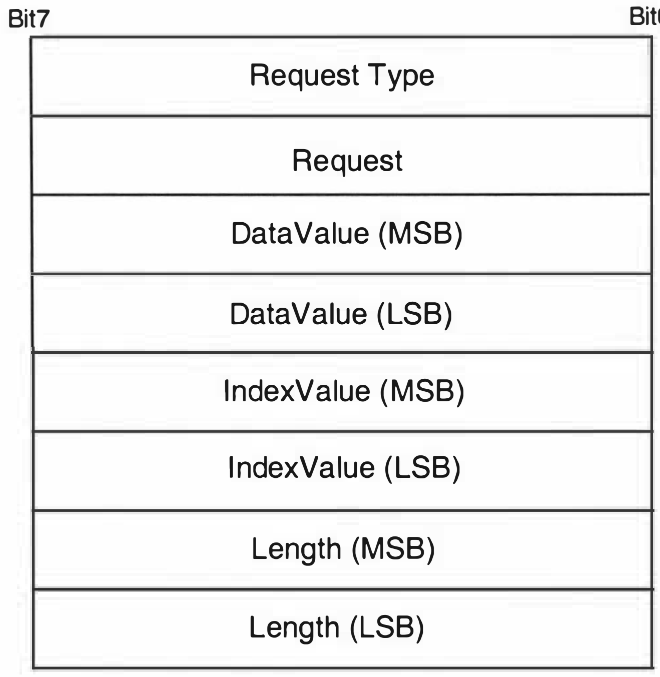


Figure 4.10 8 byte DATA0 packet within the setup phase of a control request.

The 8 byte data packet (figure 4.10) contained within the setup phase of a control request specifies what is required of the device. The first byte specifies the "Request Type" and is divided into a number of bit fields which are defined as follows:

- Bit 7 is used to indicate the direction of the optional data stage and a "0" indicates the host to device direction.
- Bits 6 and 5 are request type bits and specify if the request is a standard USB request or a user defined or class request.
- The rest of the bits are used for the optional addressing of the various components within a USB device.

The next byte is the "Request" value and indicates what function is being requested of the USB device. The "Data Value" and "Index Value" are words that the host may use to pass information to the device. The final word is the "Length" and it specifies the length of the optional data phase that follows the setup phase.

The standard requests are those that all USB devices need to support, in order to be able to be connected to the network. They are defined in chapter 9 [72] of the USB specification, and some of them are listed as follows:

- Set_Address. This request sets the USB device's address for all future accesses. When a USB device is first powered up it uses the default address of 0.

- Get_Descriptor. This request is used by the host to collect information about the device, so that it knows what it is, what it is capable of, and how it should communicate with it.

Vendor requests are requests that are specific to each device and allow a way for vendors to implement their own functions.

When a USB device is connected to the network, either through plugging it in or powering it up, it and the host go through an enumeration process which configures the device and the host for correct operation. The steps that are performed are listed as follows:

1. Firstly the host resets the USB device's interface to bring it into a known state with a device address of 0.
2. The host then reads the device descriptor using the Get_Descriptor control request in order to discover what kind of device has been attached.
3. An address is then assigned to the device using the Set_Address control request.
4. The host then reads other descriptor information to build up its knowledge about the device.
5. Using the information that the host has gathered, the host then tries to select a suitable device driver that applications will need to use in order to communicate with the device. Windows 98 onwards use "INF" files to describe what device driver to use for each type of device. These files and device drivers are usually provided with USB devices.
6. After the device driver is loaded, the USB device is brought into a "configured" state indicating that everything is functioning and that the operating system can support the device.

An application running on the host computer can then use a series of standard and vendor requests to initiate the device to perform various functions. USB is very much a master/slave arrangement, with a single master and many slave devices. The host computer is responsible for monitoring the status of the devices and for initiating any transfers if required.

Philips PDIUSBD12

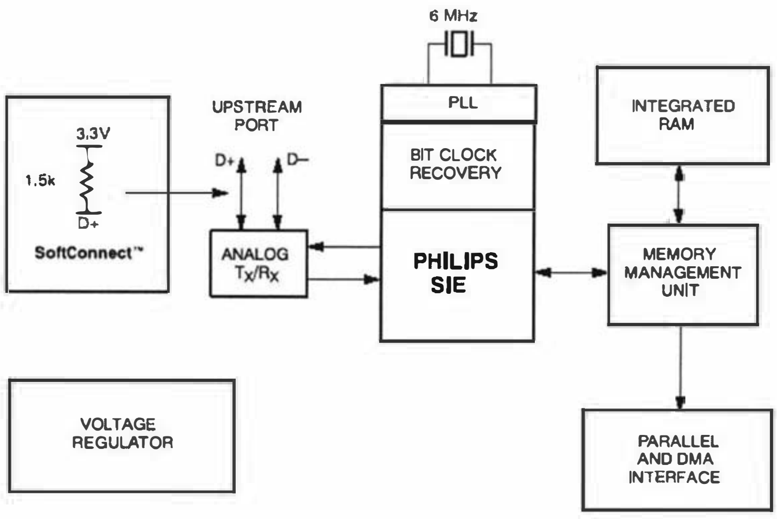


Figure 4.11 Philips PDIUSBD12. Figure taken from reference 73.

The USB interface chip chosen was the PDIUSBD12 from Philips semiconductors [73] and is shown above in figure 4.11. It is a peripheral chip that can be connected to the data bus of a microprocessor and accessed as memory. The heart of the D12 chip is the Serial Interface Engine (SIE) which implements the full USB protocol layer. It is hardwired for speed and needs no firmware intervention as it looks after many functions including: synchronisation pattern recognition, parallel/serial conversion, bit stuffing/de-stuffing, CRC checking/generation, address recognition, and handshake evaluation/generation.

The D12 also includes some internal data buffers, which help it to achieve a 1 Mbyte/s data transfer rate for bulk modes, and the interface to the host microprocessor is capable of transferring data at the rate of 2 Mbytes/s. The D12 has three endpoints, a control endpoint (0) that handles all the control requests, a generic endpoint (1) that can be used for interrupt or bulk transfers, and a main endpoint (2) that can be used for isochronous and bulk transfers. The control and generic endpoints both have a maximum packet size of 16 bytes and the main endpoint is limited to a packet size of 64 bytes.

The D12 uses a set of commands that a host microprocessor sends to it in order to control its operation. Associated with each command is a data phase, where data is transferred between the D12 and the host microprocessor. Some examples of these commands are listed as follows:

- Set Address/Enable. This is one of the initialisation commands, and is used to assign the address, and enable the endpoints of the device. The command byte is followed by a single data byte write from the host microprocessor which specifies the address.
- Select Endpoint. This command initialises an internal pointer to the start of the selected endpoint buffer. Optionally, this command can be followed by a data read which sends a status byte to the host microprocessor indicating whether the buffer is full or empty or not functioning.
- Read Buffer. This command would be sent after a select endpoint command, and is used to initiate the transfer of multiple bytes of data from an endpoint buffer to the host microprocessor.
- Write Buffer. Like the Read buffer command it is used to transfer data, but in the opposite direction.

The D12 also has a "goodlink" output that can be connected to an LED to indicate when the device is successfully enumerated and configured. The LED also blinks when data is being transferred between the host and the device. The goodlink feature has proved to be very useful when debugging the system.

PDIUSBD12 Evaluation board

Also available from Philips is a D12 evaluation system [74, 75] that consists of a circuit board with an 8051 based 8-bit microcontroller and a D12 chip mounted on it (Figure 4.12). The board behaves as a USB device and can be used to kick-start the development of custom USB devices. One of these boards was purchased to become familiar with USB hardware/software. The evaluation system came with all the source code for the board's firmware as well as a device driver, an INF file, and a simple test application for Windows98.

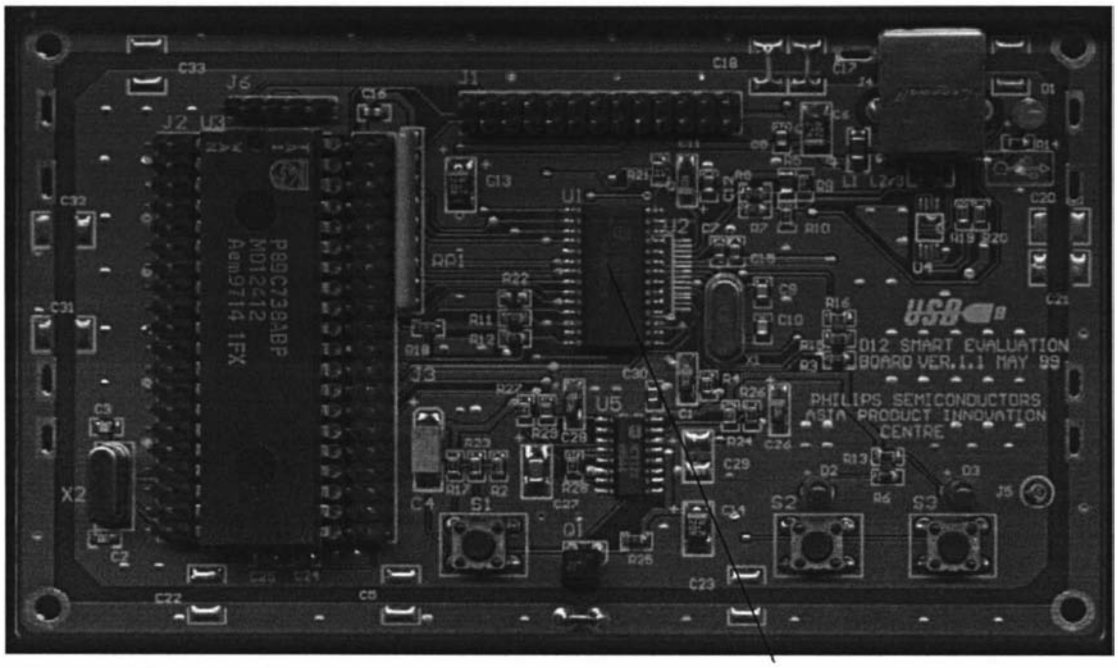


Figure 4.12 PDIUSBD12 SMART evaluation board (line pointing to chip) [74].

4.2.3 DSP/USB PCB design

After spending a considerable amount of time gaining experience with the DSP and USB devices a plan was developed for a custom DSP board that used a USB interface to communicate to a host computer. The next step was to decide what this board should have on it and how it would interface to other future boards. The resulting design features are listed below:

- JTAG interface for command converter.
- On board or off board clock.
- Extra high speed memory.
- Large bootstrap FLASH ROM.
- Buffered data/address and control lines brought out onto a backplane connector.
- Standard euro card sized board.
- Use the host port as general I/O for the USB device.
- High speed synchronous serial ports as well as other available I/O brought out onto backplane connector.

Suitable components were then selected and, after analysing all the specifications, a set of schematics were drawn up using the Protel PCB [76] CAD software. The DSP56309 device forms the heart of the board and therefore most of the design consists of providing the correct support circuits such as the reset and clock units and logic units that add to it such as memory and interfaces. The first schematic shown in figure 4.13 shows the DSP device together with the necessary reset, JTAG interface, and clock circuitry. A complete set a schematics can be found in appendix A1.

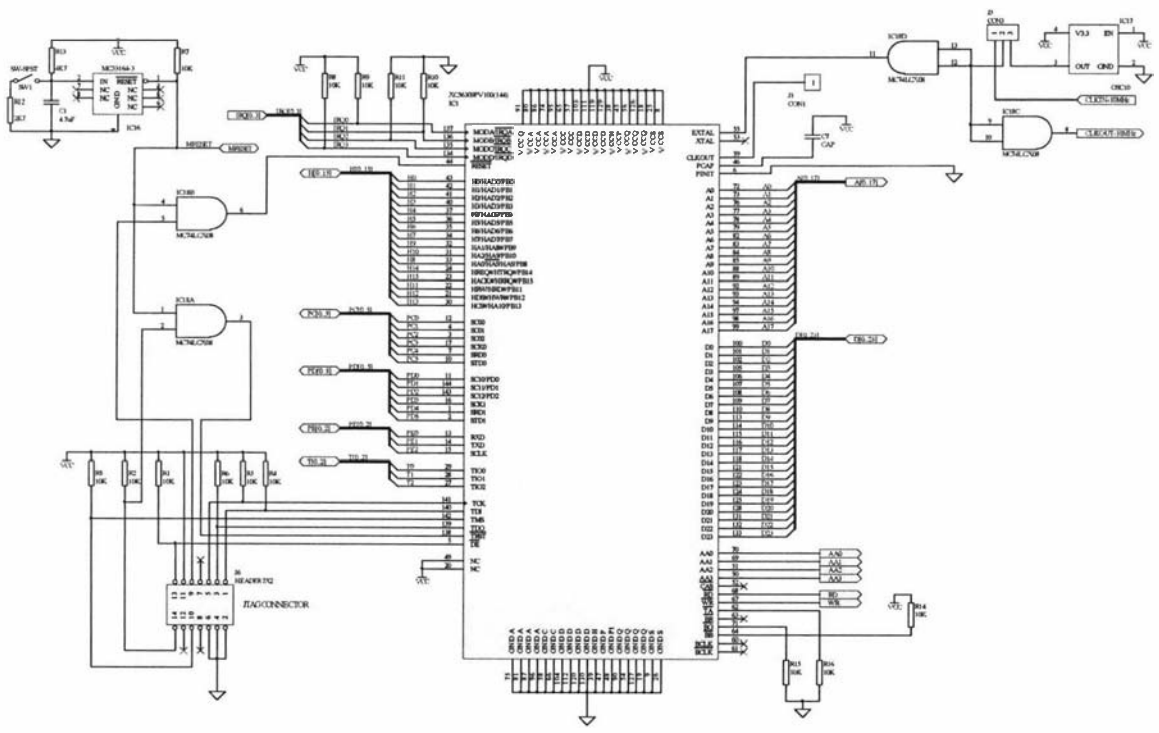


Figure 4.13 DSP section (larger version on page 165).

The reset circuit guarantees the correct start up of the DSP on power up, when the manual reset button is pressed or when an external JTAG controller forces a reset. The circuit also monitors the power supply so that if the level drops below 3 V it will force the DSP into the reset state. The JTAG interface of the DSP is brought out to a connector that matches the pin configuration of the parallel port command converter. The schematic is shown with a 10 MHz oscillator module within the clock circuit. However as the DSP uses a phase-locked loop frequency multiplier a range of different oscillator modules can be used. There is the option to use an external clock source and a buffered version of the clock is made available to the backplane connector. In the present implementation, a 25 MHz oscillator module is used with a x 4 clock multiplication factor. According to the DSP data sheet using only x 4 means that there is very little skew between the clock source and internal DSP clock. This is important later with the digital receiver when an external clock is used that must be in phase with the DSP internal clock.

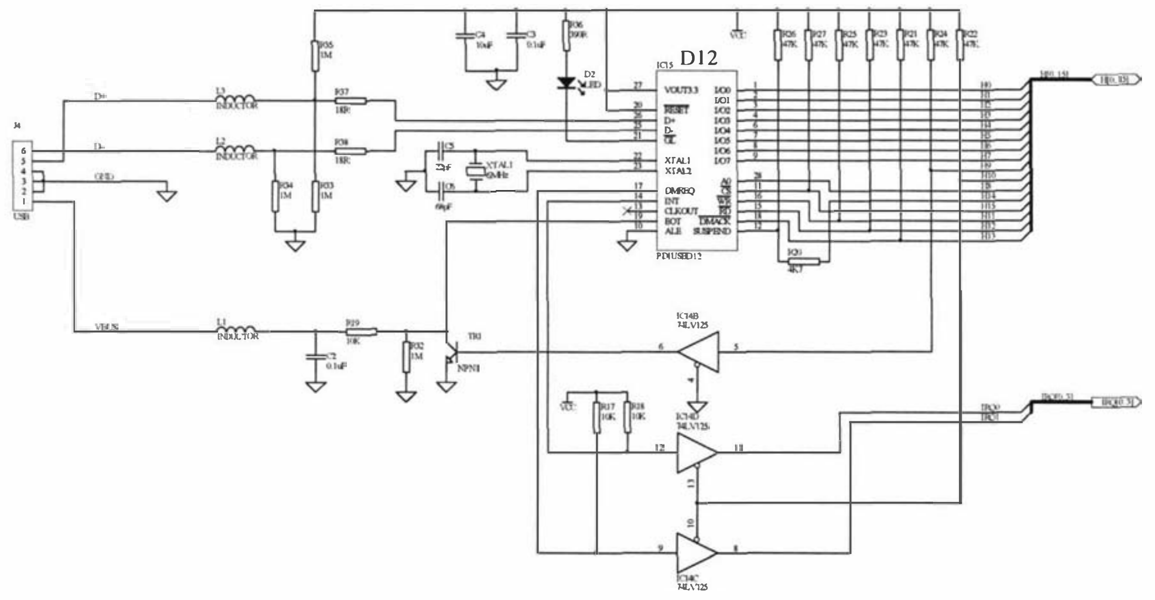


Figure 4.14 USB section (larger version on page 167).

The next schematic shown in figure 4.14 is concerned with the USB interface. The DSP host port is configured for general purpose I/O with the lower 8 bits acting as a bidirectional data bus. This is done by suitably altering the port data direction bits when required. All the other host port bits are permanently configured as outputs and are used to drive the various inputs of the D12 chip. The software running on the DSP generates the necessary control signals to allow it to communicate with the D12.

The D12, with the A0 input set high, accepts single byte commands that are written into it. This command phase is followed by a variable length data phase, where data is transferred a byte at a time between the D12 and the DSP, with A0 set to 0. In this way the D12 behaves like two successive memory locations that are addressable through the use of the A0 input pin. The D12 can gain the attention of the DSP through the use of the DSP interrupt inputs. The IRQ0 interrupt is used when a packet is transmitted or received and the D12 then requires that the DSP read the received data or write some data for the D12 to transmit. The other interrupt (IRQ1) input can be used by the D12 to signal the DSP that it wants to perform DMA between its main endpoint buffer and the DSP's memory. At the moment this feature is not used but the hardware has been put in place in case it is required in the future.

The interrupt input pins have a secondary function (mode control), such that when the DSP undergoes a reset, the status of the interrupt pins determine what operating mode the DSP will use. This feature allows hard-wired programming of the source used to boot up the DSP. A range of boot sources can be used, such as an external EEPROM or another processor connected to the host port. The D12 INT and DMREQ outputs are held low after reset and if they were directly connected to the interrupt inputs of the DSP they would interfere with the mode selection phase. Open collector tri-state buffers are therefore used to disconnect the signals until the DSP enables them. Also, during a DSP reset, the host port is tri-stated, therefore some pull-up resistors are required to prevent erroneous signals from appearing on the input pins of the D12 and to make sure that it is disabled.

If a device draws its power from the bus and sees no activity on the bus for 3 ms, it must enter a suspended state and then consume minimal power. The suspend pin of the D12 is an open collector output that is used to signal external hardware that a suspended state has been entered into. This output is normally low and is released when the D12 is suspended. The suspend pin is also an input, to allow a signal to initiate a remote wake up of the D12 when it is suspended, and therefore force it out of the suspended state. H14 (PB14) of the DSP's host port is connected to the suspend pin in order to provide the remote wake up feature if required. The 4K7 resistor in series with the suspend pin is used as protection against the fault situation in which the port output pin is high and the D12 pulls the suspend pin low.

The EOT pin of the D12 is an input pin that is used by other hardware to signal to the D12 when a DMA transfer is to be terminated. This pin is also used to monitor the 5 V VBUS supply voltage of the USB bus in order to determine if the bus exists before the D12 tries to connect to it. The D12 requires a 3.3 V supply, which is good as it is the same as the 56309 and therefore no level translation is required. The LED in series with the 390 Ω resistor is the "goodlink" indicator and the D12 uses its own clock derived from a 6 MHz crystal.

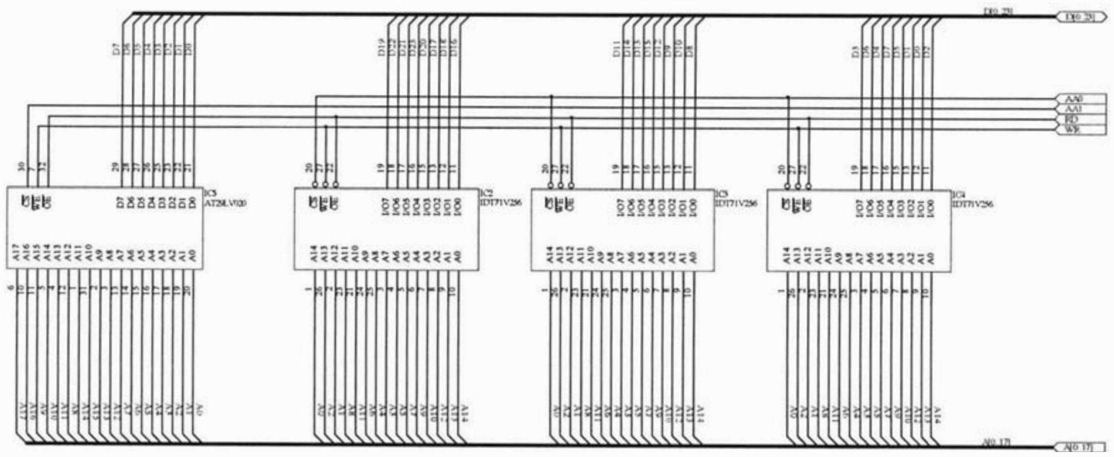


Figure 4.15 Memory section (larger version on page 168).

Figure 4.15 shows the memory section of the board which consists of some high speed 12 ns access time RAM [77] and a FLASH ROM [78]. The RAM is necessary for storing FIDs etc and is made up of three 8-bit devices to make up a 24-bit word. The internal memory of the DSP can be accessed within a single 10 ns clock cycle, but for external accesses it inserts a minimum of 1 wait state and therefore 12 ns RAM is fine for the minimum 20 ns access time. The FLASH ROM is only an 8-bit device and is used as a bootstrap ROM. When the DSP undergoes a reset, it first looks at the mode pins and if the bootstrap mode is selected it will read 8 bits at a time from the FLASH ROM and load it into the internal memory of the DSP. This feature is useful in that we only need to use a single 8-bit memory device for the bootstrap.

The backplane connector and I/O buffers are shown in figure 4.16. The backplane bus is only active when the DSP device needs to read or write to external devices. This is done to minimise the noise generated by the fast DSP board internal activity and to also minimise the power consumption. The address enable lines (AA2 & AA3) are used to activate the buffers when required and the read and write lines (RD & WR) are used set the direction of the data buffers. It should be noted that only the lower 16 bits of the address bus are provided on the backplane. This is done to minimise the number of pins needed and 16 bits of address should be adequate to support the other external devices. The access time to off board devices will be much longer due to all the extra logic involved and so the DSP must insert extra wait states. This is done by setting the appropriate DSP address attribute register.

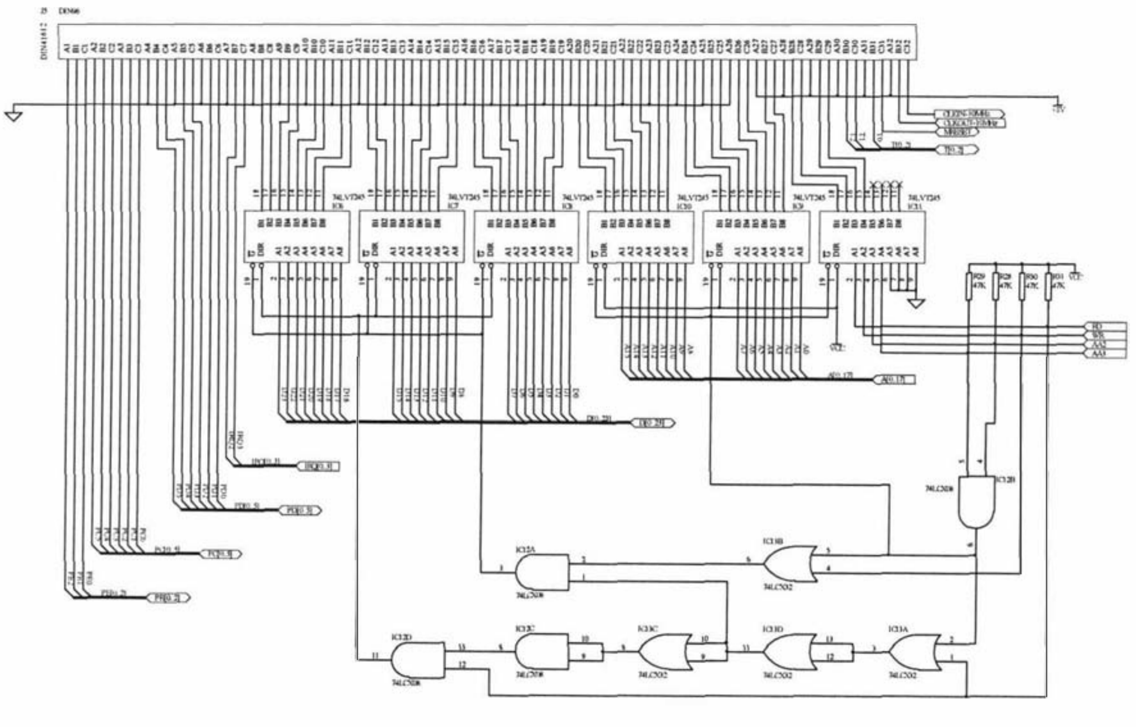


Figure 4.16 I/O section (larger version on page 166).

The final schematic shows the on-board power supply circuit and all the decoupling capacitors (figure 4.17). The supply is provided from an 8 V feed from the backplane connector which is then regulated down to 3.3 V for the DSP and all the other devices.

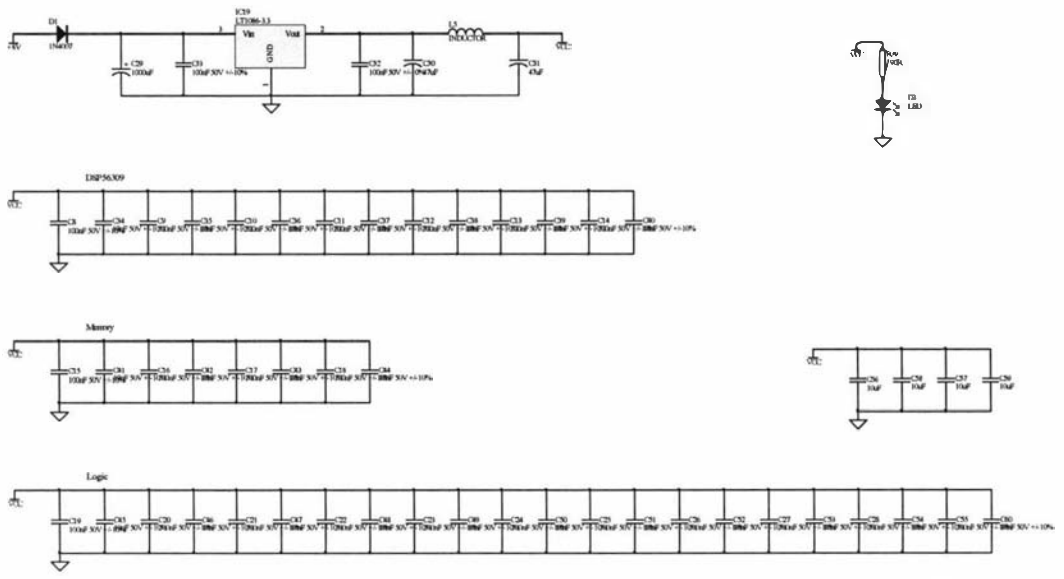


Figure 4.17 Power supply and decoupling section (larger version on page 169).

The circuit was then laid out using Protel and four layers were used. The mid layers were for the ground and 3.3V power planes. Surface mount components were used to minimise the board area and some of the components used are only available in a surface mount form. This has become a trend as through-hole components are becoming outmoded and therefore unavailable. Board assemblers also prefer that through-hole and surface mount devices are not mixed as each type requires special machines and assembly techniques. However it is often unavoidable to use a few through-hole components, especially connectors. The board is shown in figures 4.18 and figure 4.19 and measures 100 mm by 160 mm which is a standard size for a euro-card. The board was assembled using a syringe of flux, some fine solder and a range of special surface mount soldering tips for a PACE soldering station [79].

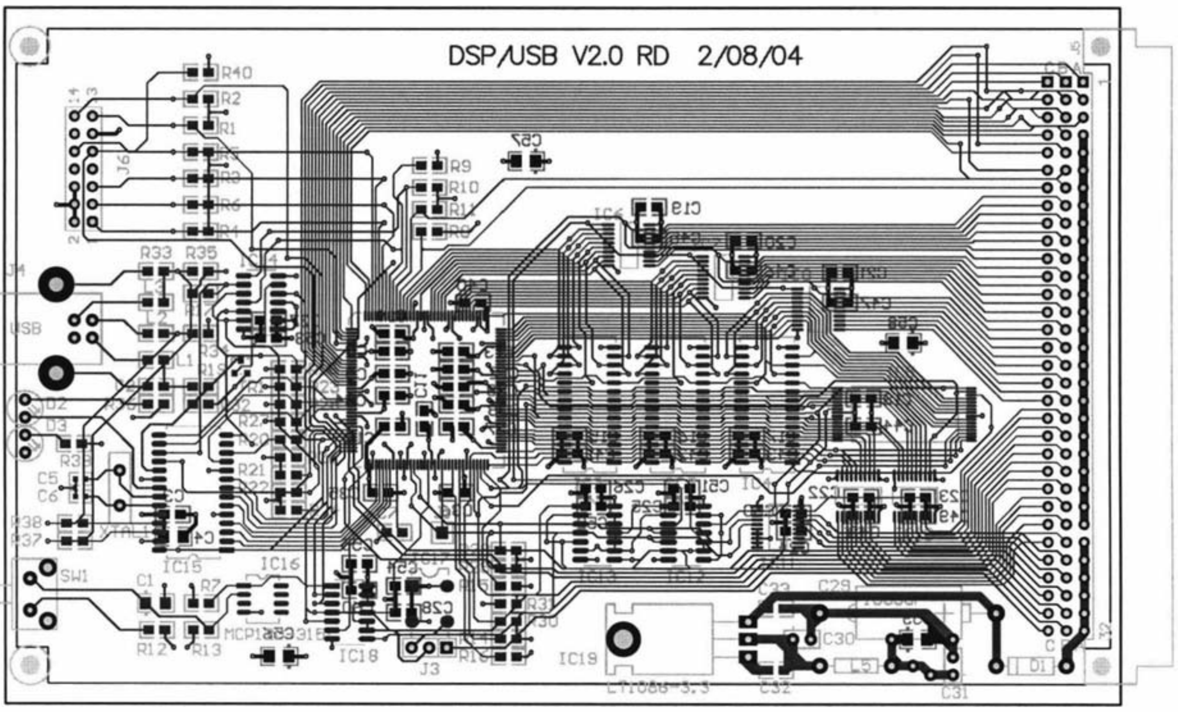


Figure 4.18 PCB layout with only the top and bottom layers shown

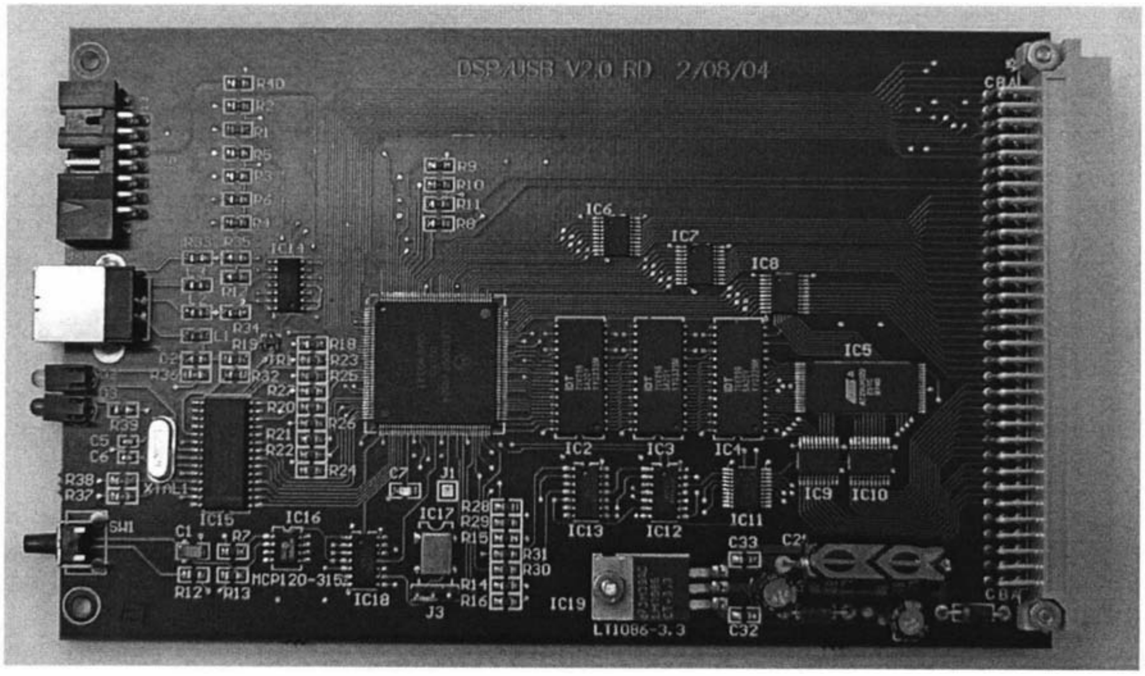


Figure 4.19 Constructed PCB.

4.3 System core software

USB development consists of two parts, firstly the design, construction and programming of a USB device, and secondly the development of a device driver and application for the host computer. Therefore the USB device development was split into two parts. The first part was to get the D12 chip working with the 56309 DSP and essentially emulate the Philips evaluation board. This meant that the Philips device driver and test application could be used to test the USB device. The next phase was to add some extra functionality to the USB device and develop a device driver and test application. Microsoft [80] provides a standard printer/scanner device driver that had all the features needed, therefore it was used instead of developing a custom driver. An API layer built on top of the WIN32 API was developed to simplify the writing of a test application. The API layer was also linked to an in-house data processing application that was then used as the user interface to the complete system. This application was primarily developed as an NMR data processing package and included a scripting language for developing macros and graphical windows.

4.3.1 DSP/USB board software

The software running on the DSP/USB board consists of two parts, the operating system that looks after the USB interface, system initialisation, memory management and command processing, and the pulse program that gets loaded and executed. The pulse program itself can be thought of as an application, as it is loaded into memory, and when the program has finished it terminates and hands back the control to the operating system.

DSP/USB operating system

A C compiler was used instead of an assembler to generate the real time embedded DSP code for the operating system for a number of reasons. The first reason is that the USB development system purchased provided all the C source code listings for the board's firmware as it was developed in C. Therefore large parts of the provided software could be used, hence saving vast amounts of time. The second reason is that the DSP is rather a difficult "beast" to program using assembly language, as it has many complicated instructions, data representations and addressing modes. The third reason for using C is that the source code is much more readable and intelligible, which is important if others want to understand how it works, in order to modify it in the future.

However, there are some disadvantages to using C. The firmware is usually much larger and therefore more program memory is required. However, the 56309 DSP device has enough memory to cope with the extra demands of the C based operating system. Another disadvantage with using C is that the code is usually slower, but the C compiler used is an optimising compiler [81], in that it tries to produce the fastest possible code. If the programmer wants maximum performance and/or the precise control over the timing of events, an assembler can be used to write selected critical routines that can be linked in and called by the C program.

To write the new operating system for the DSP board with the USB interface in C required the following:

- Firstly, setting up the C compiler so that it was compatible with the DSP board as well as configuring the C execution environment by allocating parts of the memory for the C core's variables and stack.
- Secondly, a number of I/O and interrupt routines had to be written in assembler and then linked [82] into the C environment. With C, the caller and the called routine share the responsibility of preserving the DSP registers. This is done during compilation with each looking after half the registers. This is a big trap when writing assembler routines, such as interrupt routines that call C functions, as one would normally expect the C function to preserve the registers. This makes it quite inefficient and slow for interrupt routines that call C functions as half the registers need to be saved and later retrieved every time an interrupt occurs. However, one could look through the assembly listing of the compiled C function to determine what registers it uses, and then only worry about preserving those. This would improve the interrupt handling performance of the system.
- Thirdly, the C source code had to be modified for a 24-bit big endian processor as it was originally written for an 8-bit little endian processor [74].

After many hours of debugging, the DSP/USB system was fully functional and successfully emulated the D12 evaluation board. In fact, it proved to be extremely reliable and robust, unlike the D12 evaluation board which often crashed. Testing of the board was carried out using the test application and device driver that was provided with the D12 evaluation system. This test application sends 256 byte blocks of data to the USB device, receives it back, and then checks for any errors. This is repeated in an endless loop so it can be left running to make sure no errors occur. The test application uses bulk transfers to send and receive the data, and vendor requests to initiate them. The test was left running over a day and no errors occurred.

The next phase was to add some extra functionality to the USB device to handle the execution of pulse programs and to move on to using the standard printer/scanner driver (usbscan.sys).

The DSP/USB operating system consists of a number of software modules that interact with each other in a layered hierarchical fashion as shown in figure 4.20. The source code for the modules can be found on the CD at the back of this thesis.

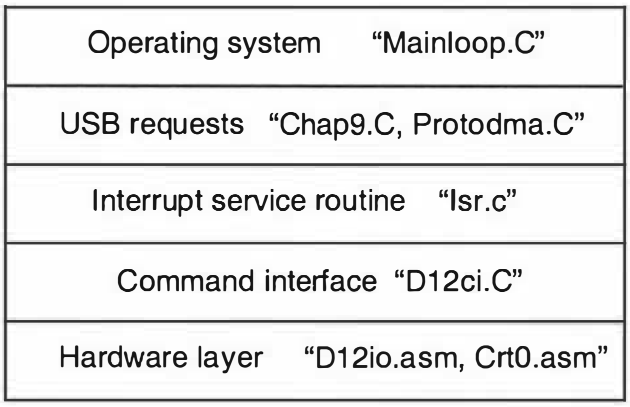

Operating system "Mainloop.C"
USB requests "Chap9.C, Protodma.C"
Interrupt service routine "Isr.c"
Command interface "D12ci.C"
Hardware layer "D12io.asm, Crt0.asm"

Figure 4.20 Layered model of the software.

The hardware layer is the interface between the hardware and the C part of the operating system. D12io.asm contains some I/O functions that the upper layer (D12ci.C) can call in order to transfer commands and data to the D12. Crt0.asm looks after the system initialisation following a reset and then passes control to Mainloop.C. The initialisation phase sets the various DSP hardware/software parameters such as the operating frequency, C variables storage area, external memory and I/O wait states, the stack and various peripherals. Crt0.asm also handles the hardware interrupt from the D12 and calls the C interrupt service routine, Isr.C, which is responsible for transferring data between memory buffers and the D12, and for the setting of flags to indicate events to the upper layers. The module D12ci.C provides a set of command interfaces, that are available to all the layers above it, that encapsulate all the functions used to access the D12. Standard USB requests are handled by the module Chap9.C, and likewise Protodma.C handles all the vendor requests. The main loop (Mainloop.C) monitors event flags and calls the appropriate subroutines to handle them. For example, when a USB request is received by the interrupt service routine, a flag is set indicating a pending request. The main loop then detects the pending request, works out if it is a standard or a custom vendor defined request, and then calls the appropriate request handler. Vendor defined requests are a way for USB developers to implement their own features. One example is the "read_write_register" request that is used to send a function request structure to the device that tells it what to perform. Some examples are: setup_bulktransfer, run_pp, get_buffer_size and get_firmware_version.

Using bulk transfers, machine code for the DSP can be loaded into its program memory and then executed through the use of the run_pp request which, after preserving the system stack etc, calls the downloaded code as a subroutine. Upon returning the run_pp request signals the host computer through the interrupt endpoint that it has finished executing the code. Normally the host computer, after requesting run_pp, would poll the interrupt endpoint. The polling itself does not interrupt the DSP as it is handled within the D12 device and therefore no problems occur with the program sequence timing. The downloaded code execution can be aborted at any stage by sending another request to the control endpoint. This results in the D12 device signalling an interrupt to the DSP, which then aborts what it was doing and restores its earlier state. This feature of being able to download and execute code has proven to be extremely useful.

To make the DSP/USB system a stand-alone device, it was necessary to get it to boot itself up after a reset. To do this, the operating system and a boot-loader routine had to be stored in the EEPROM mounted on the DSP board. This was done by using the parallel port command converter and debugger software to load the operating system software (a.cld) and all the C predefined variables into the DSP's program memory. An EEPROM programming routine (myflash.cld) that contained the code for a boot-loader was then also loaded into the DSP's program memory which, when executed, first copied the boot-loader and then the operating system and C variables data into the EEPROM. When the DSP is reset in the EEPROM bootstrap mode, it loads in the first part of the EEPROM that contains the boot-loader into a region of program memory and then starts executing it. The boot-loader then copies the operating system software and all the predefined C variables into the appropriate DSP memories and then starts executing the operating system. This technique is known as a double bootstrap. The first six bytes of the boot-loader, which are stored at the beginning of the EEPROM, tell the DSP how many bytes it has to load and where to save them. The load address also specifies where the DSP should start executing from after it has finished loading.

Getting the C and USB parts working took a considerable amount of time as many manuals and software source code listings had to be read and deciphered in order to gain enough knowledge to enable the development of the system. Also, countless hours were spent debugging the various components of the system software and hardware.

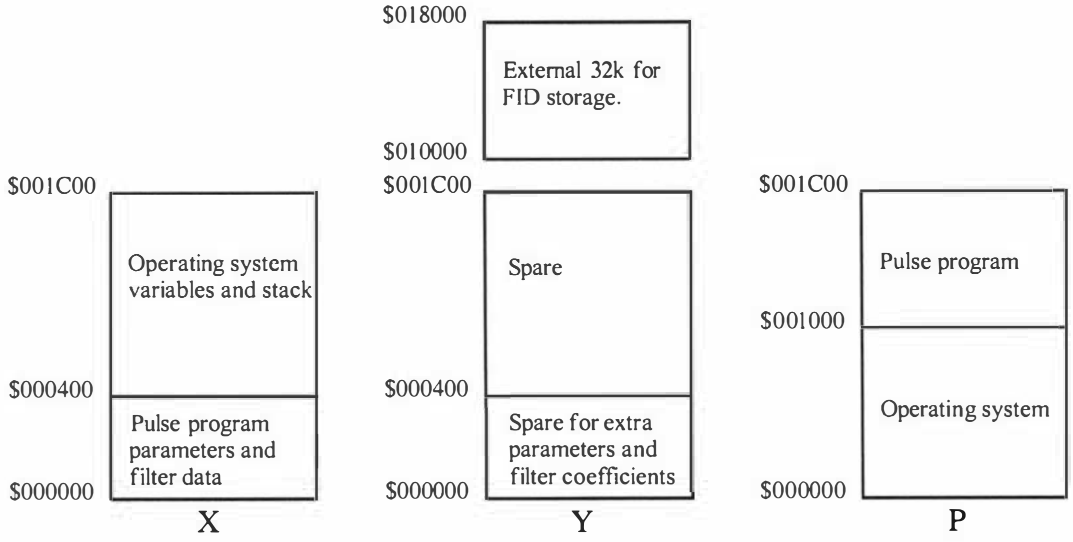
DSP pulse-program software

Pulse programs are executable code written for the DSP which implement the sequence of events that are required to perform an NMR experiment such as, for example, B_1 excitation and the subsequent sampling and storage of the FID. The pulse program is developed and stored on the host computer, and when it is required it is downloaded to the DSP board. It is called as a subroutine and therefore it must preserve the system's memory and variables. The pulse program does not necessarily have to be an NMR sequence, as being executable code it can perform any function that the user requires. This means that the system core can be used for applications other than NMR.

The pulse programs can be generated with an assembler or C compiler to produce an object file (*.cld) in the Motorola Common Object File Format (COFF) [83]. This file contains a lot of additional information such as symbol tables, so it is processed using a conversion program (Cof2ppl) to generate a file that can be downloaded to the DSP board via Prospa. This conversion program first strips off all the unnecessary information, leaving just the machine code instructions, and then six bytes containing the load address and file size are inserted at the beginning of the file. The whole process of assembling, linking and file conversion is automated using a *batch* file on the host computer, and an example batch file (robnz.bat) is listed below:

```
asm56300 -a -b -l -z robnz.asm
Cof2ppl robnz.cld
```

An individual pulse program is written for each type of experiment that the user wants to perform, but many sections are common to all experiments and most pulse programs follow a standard form. The programs always start with code that is required to set up, synchronise and precondition parts of the system. In order to be able to do signal averaging we have to guarantee that for each FID the timing of the sample points is the same. The memory requirements of the DSP are also specific to the pulse program. Figure 4.21 shows the memory usage within the X, Y and P data memories of the DSP. The FID data is stored in the external static RAM, starting at address Y:$010000 and the pulse program is loaded in program memory beginning at address P:$001000. Other parts of the memories are used to store parameters and/or tables. The parameters for the pulse programs are stored at the beginning of the X data memory and space is allocated within both the X and Y data memories for the implementation of digital filters. The USB/DSP operating system uses memory from X:$000100 onwards for its variables, stack and initialisation data and the operating system itself is stored in program memory from P:\$000000 onwards up to P:\$001000. The internal and external memory is 24 bits wide.



$018000
External 32k for FID storage.
$010000
$001C00
Operating system variables and stack
$000400
Pulse program parameters and filter data
$000000
X
$001C00
Spare
$000400
Spare for extra parameters and filter coefficients
$000000
Y
$001C00
Pulse program
$001000
Operating system
$000000
P

Figure 4.21 DSP board memory allocation.

To change a parameter for a pulse sequence one only needs to change the contents of a memory location rather than having to reassemble and download a new pulse sequence. This is essential as often we need to alter a parameter, such as the echo time or gradient amplitude, between scans.

4.3.2 Windows 2000/XP Device driver and API

A custom vendor specific driver is usually required for a USB device, but some standard drivers have been provided that can be used to support particular classes of devices such as scanners, printers and digital cameras. Device drivers are not the easiest things to develop. However there is very good developer support available from Microsoft [84] and others [85], who provide driver development kits (DDKs) and wizards. Even with all this support it still much easier to use one of the standard provided drivers and so the DSP/USB system was designed to mimic a still image class device so that the "usbscan.sys" driver provided by Microsoft could be used. Usbscan.sys supports a single control endpoint, along with multiple bulk and interrupt endpoints. Using standard drivers also has the advantage that whenever Microsoft updates their operating system they will most likely provide a new version of the driver and therefore no changes will be required to use my DSP/USB system.

Windows uses a series of kernel mode (trusted) drivers organised in a hierarchical layered fashion as shown in figure 4.22.

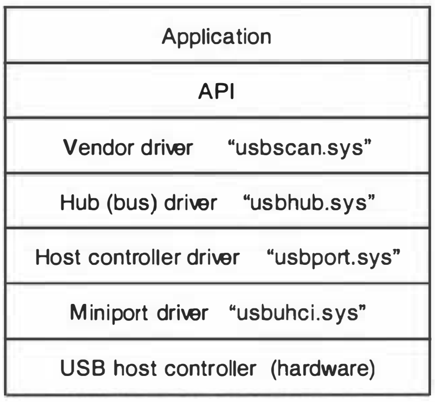

Application
API
Vendor driver "usbscan.sys"
Hub (bus) driver "usbhub.sys"
Host controller driver "usbport.sys"
Miniport driver "usbuhci.sys"
USB host controller (hardware)

Figure 4.22 Layered model of host computer software.

Applications can communicate with the uppermost driver (usbscan.sys) through a set of standard C functions provided by the windows operating system, known as the Win32 Application Programmers Interface (API). Some of these functions that can be called by the application are:

- CreateFile. This is used to obtain a unique identifier (handle) to the device that is used for all subsequent accesses.
- ReadFile. This is used to retrieve data from a device endpoint using bulk transfers.
- WriteFile. This is used to send data to a device endpoint using bulk transfers.
- DeviceIOControl. This function, together with some predefined I/O control codes and data structures, is used to access the control and interrupt endpoints.

The Windows API converts these function calls into an I/O request packet (IRP), and then passes them down to the uppermost device driver. Each IRP contains an I/O control code, which describes what operation is required. The task of the device driver is to process the IRP by calling a function that matches the I/O control code. The called function then has the task of generating a series of USB request blocks (URBs) that are passed down to the USB bus driver (usbhub.sys). The URBs also use control codes to indicate what function is being requested. Custom I/O control codes can be used within the IRPs in order to implement special features, but the URBs are standard and are defined by the USB bus driver. Some URB functions that can be requested from usbhub.sys are listed as follows:

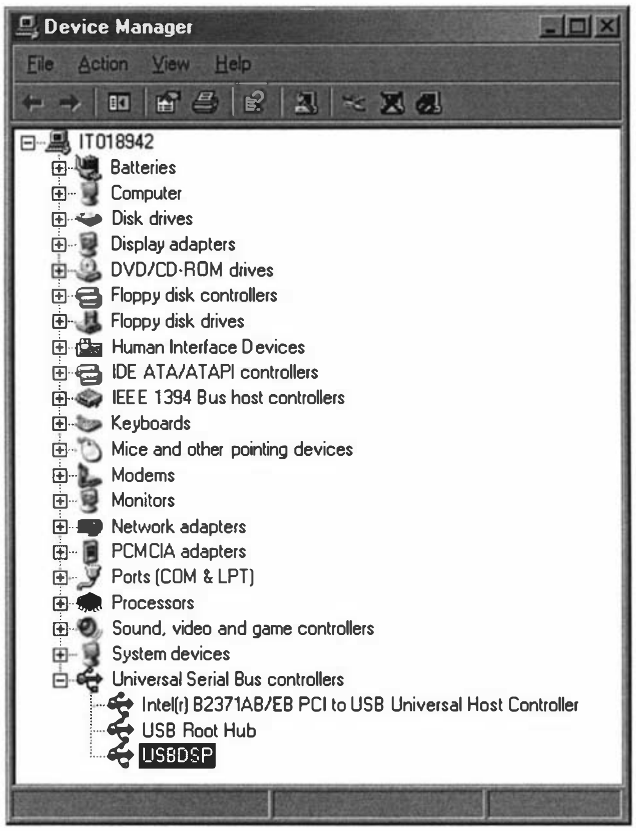
1. GET_DESCRIPTOR_FROM_DEVICE. This function retrieves the device descriptor from a USB device.
2. VENDOR_DEVICE. This function sends a vendor control request.
3. BULK_OR_INTERRUPT_TRANSFER. This function performs a bulk or interrupt transfer.

INF files

When a device is installed for the first time, an "INF" file is used to tell the windows operating system what driver to load for that device. The device is registered with the windows operating system so that it knows how to handle it now and in the future. Later, when a device is plugged in, Windows interrogates the device using a special address and some standard protocol commands. Information about the device is extracted and then used to recognise the device so that the correct driver is setup to handle the device. The first bit of information it gets from the device is the vendor and product identifiers which Windows then uses to search through its device database within the registry for a match. The registry device database contains information about what driver to use to support the device and some other parameters that are specific to the device. Each device that is designed to run under the Win32 platform must have a device class. Each device class has a class key that uniquely identifies the class (GUID). Whenever a new device is installed, the GUID is used to determine the sub-key under which to write registry information.

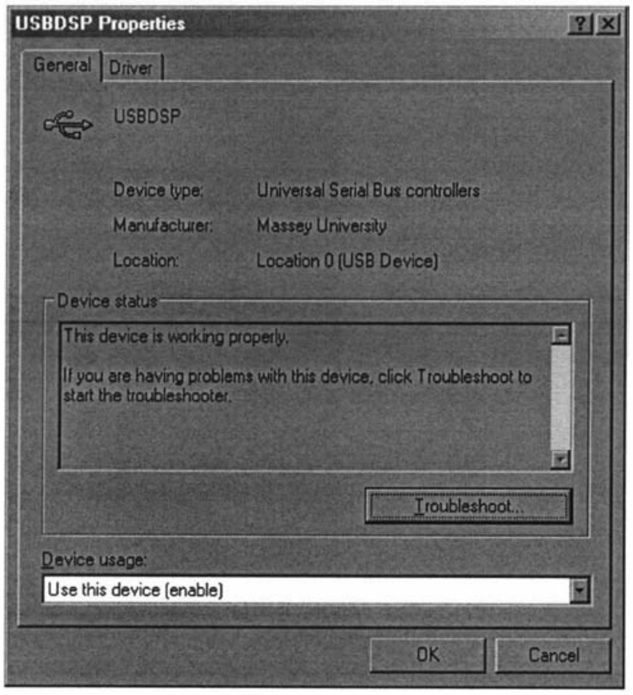
A wizard, that was provided with the WINXP DDK 2600 [84] called "GenINF", was used to produce an INF file for the DSP/USB board. The vendor and product identifiers were kept the same as the Philips development board but, if needed, others can be obtained. A reduced version of the INF file is shown below with some comments inserted. A full version of the INF file is included on the CD that is at the back of this thesis.

```
;                       A:\robusb.inf
;
;        Created by GenINF.

[Version]
Signature = "$Windows NT$"
Class=USB
ClassGUID={36fc9e60-c465-11cf-8056-444553540000} // Still image class GUID
Provider=%Massey%
CatalogFile=robcat.cat
DriverVer= 8/1/2004

[DestinationDirs]
ROBDSP/1.Files.x86_12 = 12

[SourceDisksNames.x86]
0=%Desc_x860%

[SourceDisksFiles.x86]
usbscan.sys=0,,

[Manufacturer]
%Massey%=Massey

[Massey]
%ROBDSP/1Desc%=ROBDSP/1_Inst,USB\VID_0471&PID_0666   //Vendor and product identifiers

[ROBDSP/1_Inst.ntx86]
CopyFiles = ROBDSP/1.Files.x86_12

[ROBDSP/1_Inst.ntx86.Services]
AddService = usbscan,0x00000002,ROBDSP/1_Service_Instx86, //This modifies the registry

[ROBDSP/1_Service_Instx86]
ServiceType = %SERVICE_KERNEL_DRIVER%
StartType = %SERVICE_DEMAND_START%
ErrorControl = %SERVICE_ERROR_IGNORE%
ServiceBinary = %12%\usbscan.sys

[ROBDSP/1.Files.x86_12]            //The device driver file to use for ROBDSP
usbscan.sys

[Strings]

;  *******Localizable Strings*******
Massey= "Massey University"                //Device company name
Desc_x860= "Massey University USBdrivers"
               ROBDSP/1Desc= "USBDSP"                     //Device name
```



```
IT018942
    Batteries
    Computer
    Disk drives
    Display adapters
    DVD/CD-ROM drives
    Floppy disk controllers
    Floppy disk drives
    Human Interface Devices
    IDE ATA/ATAPI controllers
    IEEE 1394 Bus host controllers
    Keyboards
    Mice and other pointing devices
    Modems
    Monitors
    Network adapters
    PCMCIA adapters
    Ports (COM & LPT)
    Processors
    Sound, video and game controllers
    System devices
    Universal Serial Bus controllers
        Intel(r) 82371AB/EB PCI to USB Universal Host Controller
        USB Root Hub
        USBDSP
```

Figure 4.23 Windows' device manager control panel showing the installed USBDSP device.



USBDSP

Device type: Universal Serial Bus controllers
Manufacturer: Massey University
Location: Location 0 (USB Device)

Device status

This device is working properly.

If you are having problems with this device, click Troubleshoot to start the troubleshooter.

Device usage: Use this device (enable)

Figure 4.24 USB/DSP device properties panel.

Before one can communicate with a device, a unique device "handle" needs to be obtained so that the windows operating system knows with which device you are trying to communicate.

When the DSP/USB device was installed, the Windows properties panel showed that the device was at location "0" (figures 4.23 and 4.24). This meant that it was the first device that had been registered within the system registry to the device driver usbscan.sys. Multiple devices can use the same driver and, if this is the case, one would then have to interrogate the devices at the various locations to determine which was the correct one to use. This can be done by looking at the product and vendor IDs.

To get a handle to the first device we simply append the location number to the device driver name and pass that to a "Createfile" function. Some additional backslashes are included to inform windows that a physical device is being selected.

```
char name[MAX_PATH];

strncpy(name,"\\\\.\\Usbscan0", 15);

HANDLE hdevice;

hdevice = CreateFile(name, GENERIC_READ | GENERIC_WRITE,0,NULL,OPEN_EXISTING,0,NULL);
```

To communicate one also needs to know what pipes are available and what their respective addresses are. For the DSP/USB system the pipes have been defined as:

Control read:	Control pipe	D12 Endpoint index 0
Control write:	Control pipe	D12 Endpoint index 1
Interrupt read:	Pipe 0	D12 Endpoint index 2
Interrupt write:	Pipe 1	D12 Endpoint index 3
Bulk read:	Pipe 2	D12 Endpoint index 4
Bulk write:	Pipe 3	D12 Endpoint index 5

To address a specific pipe, one simply appends the pipe address to the end of the device name, eg `\\\\.\\Usbscan0\\2` would select the bulk read pipe. By default, the control pipe is also always selected and associated with the individual pipe handle.

To perform a bulk read or write we first notify the USB device using the "DeviceIOcontrol" function of the task we want to perform and then we read or write data to/from the device using the standard readfile/writefile functions. Some C source code that performs a bulk data read transfer is shown below.

```
strncpy(dname, name, sizeof(name));
strcat(dname, "\\2");
printf("\n\nFull name is: %s\n", dname);

hdevice = CreateFile(dname, GENERIC_READ | GENERIC_WRITE,0,NULL,OPEN_EXISTING,0,NULL);

                if(hdevice == INVALID_HANDLE_VALUE)
                      {
                      printf("\nUnable to open device - error %d\n",GetLastError());
                      }
                else
                {
                      data = ReadData(hdevice,data);
                      CloseHandle(hdevice);
                }
```

```
ReadData(HANDLE hdevice, char *rdata)
{
   long nBytes;
   short result;

   ioRequest.uAddressL = 0x0000;      // address 010000 within DSPUSB device
   ioRequest.bAddressH = 0x01;
   ioRequest.uSize = transfer_size;           // number of bytes to transfer
   ioRequest.bCommand = 0x31;   // read from Y memory

   ioBlock.uOffset = 0;
   ioBlock.uLength = sizeof(ioRequest);
   ioBlock.pbyData = (PUCHAR)&ioRequest;
   ioBlock.uIndex = 0x0471;           // set up DMA command

   result = DeviceIoControl(hdevice,
                            IOCTL_WRITE_REGISTERS,
                            (LPVOID)&ioBlock,
                            sizeof(IO_BLOCK),
                            (LPVOID)NULL,
                            0,
                            (LPDWORD)&nBytes,
                            (LPOVERLAPPED)NULL);
   if(result == false)
   {
      printf("\nSetup DMA request failed!- error %d\n",GetLastError());
      return(NULL);
   }

   printf("USB IO control command sent\n");

   result = ReadFile(hdevice,
                     (LPVOID)rdata,
                     (DWORD)transfer_size,
                     (LPDWORD)&nBytes,
                     (LPOVERLAPPED)NULL);

   if(result == false)
   {
      printf("\nRead failed!- error %d\n",GetLastError());
      return(NULL);
   }

   printf("USB, Number of bytes received = %d\n", nBytes);

   return(rdata);
}
```

The DeviceIoControl function sends a command request (IOCTL_WRITE_REGISTERS) and a data structure (ioBlock) to the USB device to specify a bulk data transfer of n bytes from/to memory starting at address 0x010000. The I/O control code: IOCTL_WRITE_REGISTERS is a standard Windows USB Still image I/O control code that is used to write to the devices registers using the control pipe. It is a vendor request, which together with ioBlock.uIndex = 0x0471, is interpreted as a read_write_register request requesting a bulk transfer. ioRequest.bCommand = 0x31 specifies that it is read from Y memory request. The "ReadFile" function is then used to transfer the data from the device using pipe 2. The maximum bulk transfer size is 768 words, so if larger amounts of data need to be transferred, multiple requests are required. This can be done by simply altering the address of the starting memory location for each request. This bulk data transfer technique is referred to as a "DMA" transfer as it transfers large amounts of data between the Windows system and the USB device with minimal application intervention. In other words, most of what occurs is transparent to the high level application and it "thinks" that it is simply transferring data between itself and the DSP/USB board's memory.

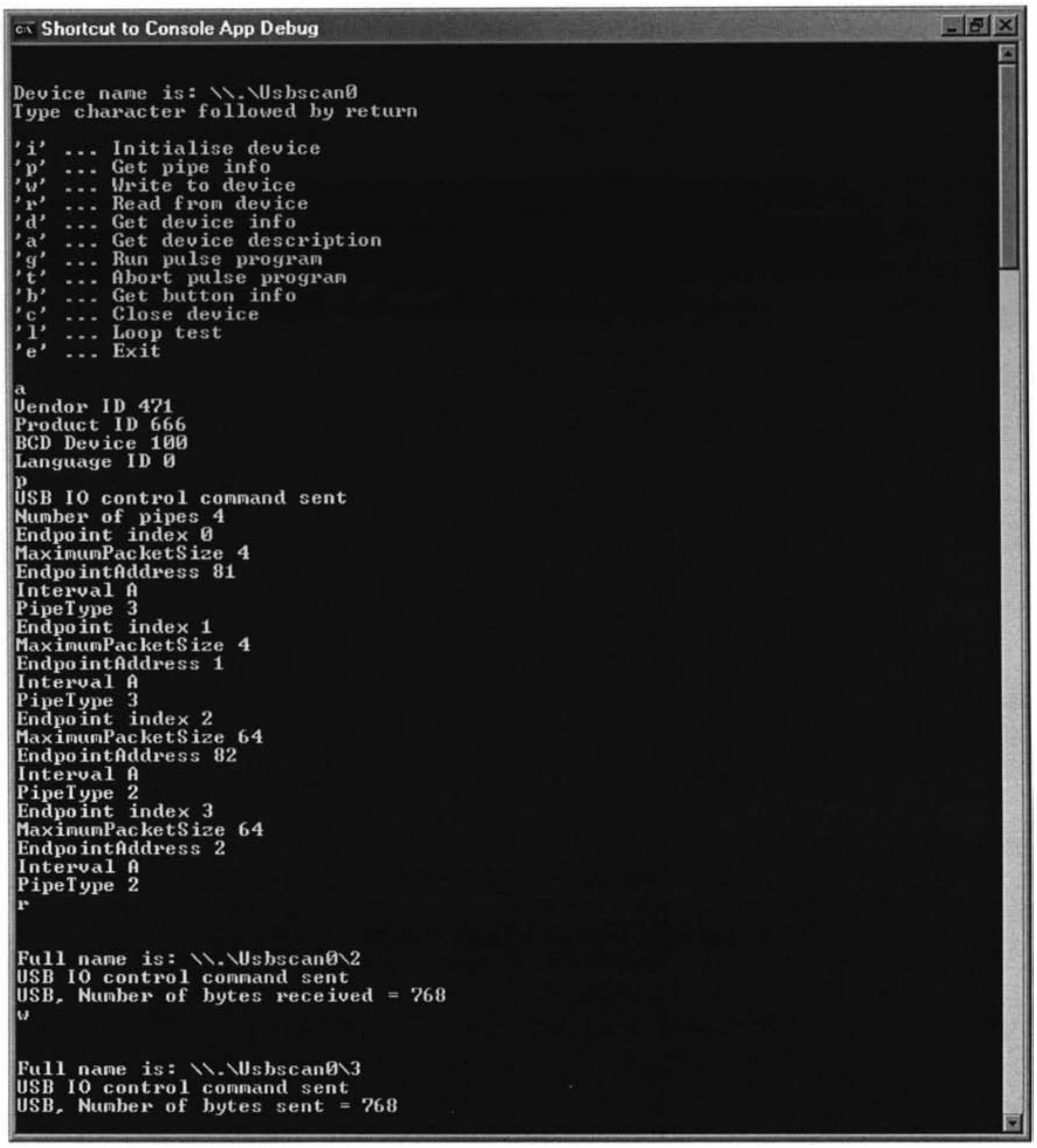
To perform a bulk write to Y memory we do the same as for a read except that we set the ioRequest.bCommand to 0x30 and use pipe 3 to send the data. This is shown below.

```
strncpy(dname, name, sizeof(name));
strcat(dname, "\\3");
printf("\n\nFull name is: %s\n", dname);

hdevice = CreateFile(dname, GENERIC_READ | GENERIC_WRITE,0,NULL,OPEN_EXISTING,0,NULL);

                 if(hdevice == INVALID_HANDLE_VALUE)
                       {
                       printf("\nUnable to open device - error %d\n",GetLastError());
                       }
                 else
                 {
                       for(int i = 0; i < transfer_size; i++)
                              data[i] = i;
                       WriteData(hdevice,data);
                       CloseHandle(hdevice);
                 }

WriteData(HANDLE hdevice, char *data)
{
   long nBytes;
   short result;

   ioRequest.uAddressL = 0x0000;      // address 010000
   ioRequest.bAddressH = 0x01;
   ioRequest.uSize = transfer_size;           // number of bytes to transfer
   ioRequest.bCommand = 0x30;   // write to Y mem

   ioBlock.uOffset = 0;
   ioBlock.uLength = sizeof(ioRequest);
   ioBlock.pbyData = (PUCHAR)&ioRequest;
   ioBlock.uIndex = 0x0471;           // set up DMA command


   result = DeviceIoControl(hdevice,
                            IOCTL_WRITE_REGISTERS,
                            (PVOID)&ioBlock,
                            sizeof(IO_BLOCK),
                            (LPVOID)NULL,
                            0,
                            (LPDWORD)&nBytes,
                            (LPOVERLAPPED)NULL);

   if(result == false)
   {
      printf("\nSetup DMA request failed!- error %d\n",GetLastError());
      return(false);
   }

   printf("USB IO control command sent\n");

   result = WriteFile(hdevice,
                      (LPVOID)data,
                      (DWORD)transfer_size,
                      (LPDWORD)&nBytes,
                      (LPOVERLAPPED)NULL);

   if(result == false)
   {
      printf("\nWrite failed!- error %d\n",GetLastError());
      return(false);
   }

   printf("USB, Number of bytes sent = %d\n", nBytes);

   return(true);
}
```

To read from the interrupt pipe, we select pipe zero and use the control code "IOCTL_WAIT_ON_DEVICE_EVENT". This causes windows to poll the USB device every millisecond to see if it has any data in its interrupt out buffer. Once some data has been obtained, Windows stops polling and returns from the DeviceIoControl function with the data. The code to perform this is shown below and is useful for monitoring the status of a USB device. One example of this is when Windows is trying to determine when a pulse program running on the DSP/USB device has finished executing. The DSP/USB device operating system signals Windows when pulse program execution is finished by placing data into the interrupt out buffer.

```
strncpy(dname, name, sizeof(name));
strcat(dname, "\\0");
printf("\n\nFull name is: %s\n", dname);

hdevice = CreateFile(dname, GENERIC_READ | GENERIC_WRITE,0,NULL,OPEN_EXISTING,0,NULL);

                if(hdevice == INVALID_HANDLE_VALUE)
                      {
                      printf("\nUnable to open device - error %d\n",GetLastError());
                      }
                else
                {
                   data = GetButton(hdevice, data);
                      printf("data =: %d\n", data[0]);
                      CloseHandle(hdevice);
                 }

GetButton(HANDLE hdevice, char *rdata)       //read the button state using the
interrupt
{
   long nBytes;
   short result;
   const long intpack_size = 4;

   result = DeviceIoControl(hdevice,
                               (DWORD)IOCTL_WAIT_ON_DEVICE_EVENT,
                               NULL,
                               0,
                               (PVOID)rdata,
                               (DWORD)intpack_size,
                               (LPDWORD)&nBytes,
                               (LPOVERLAPPED)NULL);

   if(result == false)
   {
      printf("\nRead failed!- error %d\n",GetLastError());
      return(NULL);
   }

   printf("USB, Number of bytes received = %d\n", nBytes);

   return(rdata);
}
```

All the C source code for the previous few examples are included on the CD and are part of an application that was written to test the device and data transfer methods. This was a simple console application written using a CodeWarrier compiler [86]. A typical screen shot is shown below in figure 4.25. Here we can see the device and pipe information as well as an example of a bulk read and write.



```
Device name is: \\.\Usbscan0
Type character followed by return

'i' ... Initialise device
'p' ... Get pipe info
'w' ... Write to device
'r' ... Read from device
'd' ... Get device info
'a' ... Get device description
'g' ... Run pulse program
't' ... Abort pulse program
'b' ... Get button info
'c' ... Close device
'l' ... Loop test
'e' ... Exit

a
Vendor ID 471
Product ID 666
BCD Device 100
Language ID 0
p
USB IO control command sent
Number of pipes 4
Endpoint index 0
MaximumPacketSize 4
EndpointAddress 81
Interval A
PipeType 3
Endpoint index 1
MaximumPacketSize 4
EndpointAddress 1
Interval A
PipeType 3
Endpoint index 2
MaximumPacketSize 64
EndpointAddress 82
Interval A
PipeType 2
Endpoint index 3
MaximumPacketSize 64
EndpointAddress 2
Interval A
PipeType 2
r


Full name is: \\.\Usbscan0\2
USB IO control command sent
USB, Number of bytes received = 768
w


Full name is: \\.\Usbscan0\3
USB IO control command sent
USB, Number of bytes sent = 768
```

Figure 4.25 Test application.

To make it easier for those who want to write their own user interface applications, the Win32 API functions, together with all the other necessary information, were combined into a set of higher level, more user friendly functions. Some examples of these higher level functions are:

- DownLoad_program.
- Run_program.
- Abort_program.
- Wait_on_program.
- Read_data.
- Write_data.

Using these functions means that the application programmer does not need to know anything about USB. All they are concerned with is transferring data etc to and from the USB device's memory. In a sense they communicate with the uppermost layer of the USB system software and all that happens in between is transparent. This is the idea of modern communications where layers are used to hide the complexity from the applications. This is shown in figure 4.26 where a virtual link is formed between the upper layers of the communications protocol stack. The application does not know about the hardware and "thinks" that it is communicating directly with the DSP/USB board's operating system. In reality, when the application calls one of the functions provided by the API, commands are sent down through the layers and across to the DSP/USB board, and then the DSP/USB board's operating system takes the appropriate action. With this arrangement the host laptop computer operates as a master and the DSP/USB board as a slave, however the laptop and the DSP/USB board each operate independently using their own software.

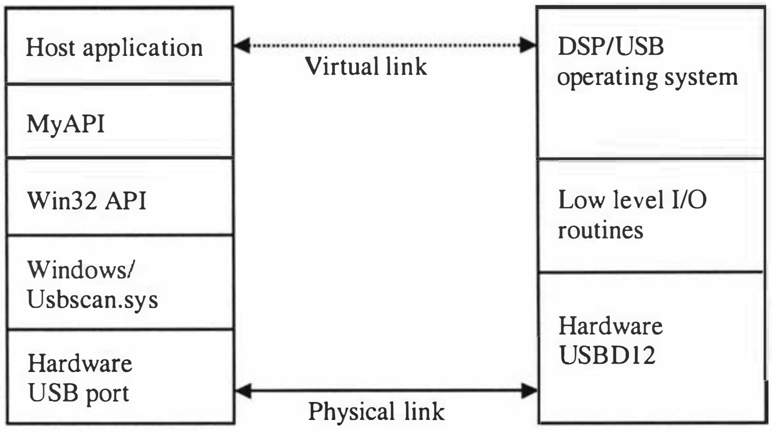


Figure 4.26 Communications protocol stack.

Each layer within the protocol stack only communicates with neighbouring layers. Using a protocol stack model is helpful when developing software for two systems that need to communicate, as it clearly defines what each part of the system has to do. Being modular makes it a lot easier when modifications are required.

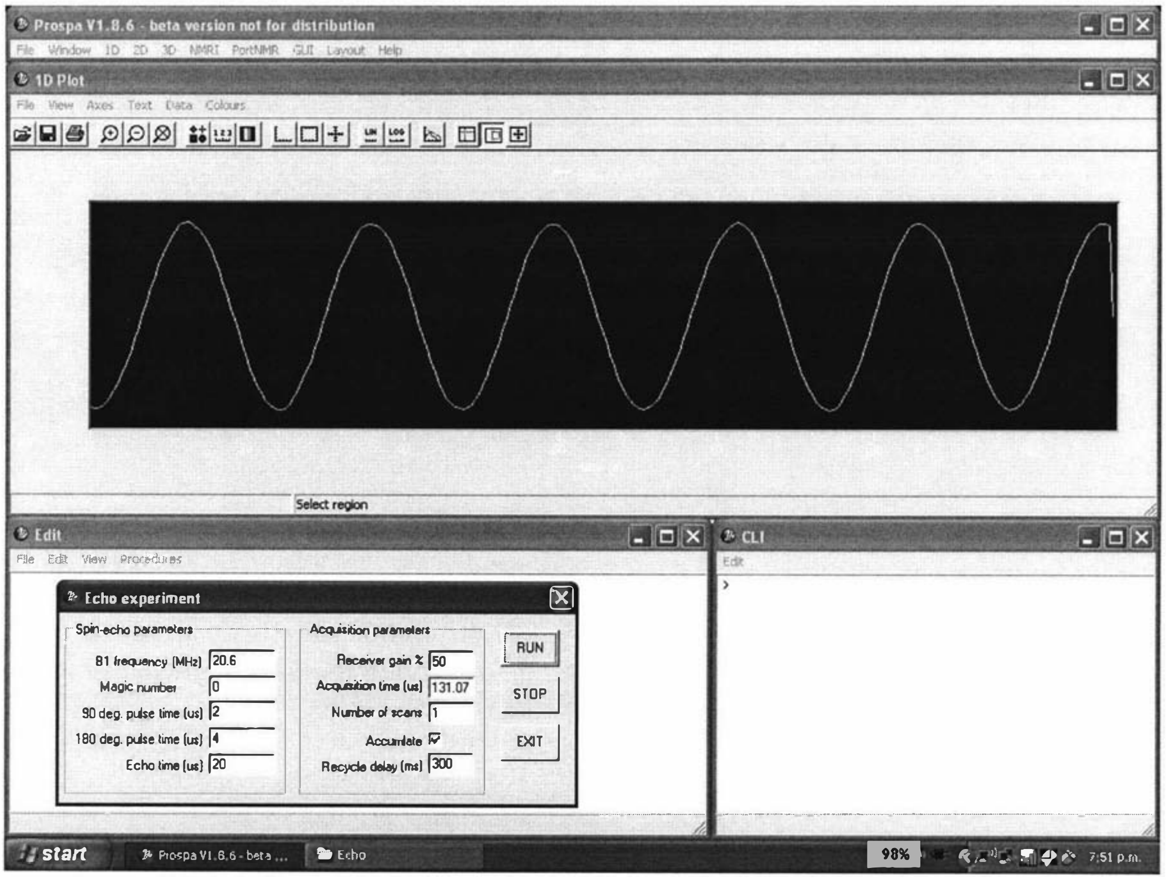
The application layer on the top is the user's application, and in this case is either the test program (figure 4.25) or a NMR data processing package called "Prospa" that was developed by Dr Craig Eccles [87] a number of years ago and was enhanced to include data acquisition and control. The functions provided by MyAPI can be thought of as a new layer, but are really on the same level as the user's application.

One problem of using protocol stacks is the often large numbers of layers involved. This layered approach often requires more code to be written which leads to an increase in the execution time. Flexibility comes at a cost.

4.3.3 Prospa

Prospa is a program that was initially developed for NMR data processing on Macintosh and UNIX platforms, but has now been ported over to the PC platform and extended with control capabilities. This allows us to control, capture and analyse data within one software package. As Prospa is developed in-house it can be modified and expanded easily to suit many applications, including ones other than NMR. The main features of Prospa are the built-in scripting language and the user-programmable graphical window interfaces. Figure 4.27 below is a screen dump from the laptop with Prospa running and shows how the user interface can be divided up into a number of sections. The individual windows can be resized and moved to any position, and other window interfaces can be generated, if desired, using a window layout script. In this way a window can be created that may contain some user defined buttons and a plotting area for displaying graphs.

The 1D plot window appears at the top of the screen, just below the application's main menus, and is used to display the FID data or the corresponding spectrum. Some buttons are included that allow the saving and retrieval of data, the selecting and zooming of various regions and the changing of various plot parameters.


Prospa V1.8.6 - beta version not for distribution
File Window 1D 2D 3D NMRI PortNMR GUI Layout Help
1D Plot
File View Axes Text Data Colours
Select region
Edit
File Edit View Procedures
CLI
Edit
Echo experiment
RUN
STOP
EXIT
start
Prospa V1.8.6 - beta ...
Echo
98%
7:51 p.m.

Spin-echo parameters

B1 frequency (MHz)	20.6
Magic number	0
90 deg. pulse time (us)	2
180 deg. pulse time (us)	4
Echo time (us)	20

Acquisition parameters

Receiver gain %	50
Acquisition time (us)	131.07
Number of scans	1
Accumlate	✓
Recycle delay (ms)	300

Figure 4.27 Screen dump of laptop with Prospa running.

The command line interface (CLI) window is shown on the bottom right hand side. It is used when only a single line command is required to be executed. An example is the "ft" command, which can be used to perform a Fourier transform of the data displayed in the 1D plot window. Prospa has many such commands (or functions) contained within its function library. The library itself can be easily expanded as it allows C code modules to be easily linked into it via the use of Dynamic Linking Libraries (DLLs). The functions in the application interface layer (MyAPI) have been included within Prospa's function library and therefore can be executed from the CLI.

An editor window is shown on the bottom left hand side of the screen dump and is used for editing text files such as a macro. Prospa provides a macro scripting language that can be used to automate commonly used command sequences. All the functions contained within Prospa's function library can be used within macros. For-Next loops and If-Then-Else switches are also included to provide a full programming-like language. A graphical control interface for an experiment is placed near the middle of the editor window area and demonstrates some of the user interface capabilities of Prospa.

Prospa also contains a "help" system that can be accessed from the main application menu. This application menu is also configurable so that users can implement their own menus and commands.

5.0 Digital transceiver

5.1 Introduction

The system core on its own is of little use. What is also required are some modules that can provide the connection to the outside world. In the case of NMR we need a transceiver module that can handle all the radio frequency signals associated with the probe. This is shown below in figure 5.0. The only interfaces that exist on the system core DSP/USB board are digital and therefore the transceiver module must use digital signals at this interface and radio frequency analogue signals at the probe interface. So obviously some D-A and A-D conversion processes must take place within the transceiver. Today the trend is to place these conversion processes as close as possible in the signal chain to the outside world. This reduces the amount of analogue circuitry within the signal chain as more of the processing is done digitally. This is attractive as analogue circuitry often takes up a lot of board real estate and is also prone to DC offsets, gain errors, interference, distortion, and drift. The approach taken for the transceiver design has been to try and implement this new trend by taking advantage of some new technologies. One example in particular is the new technologies that have been developed for the cellular phone industry.

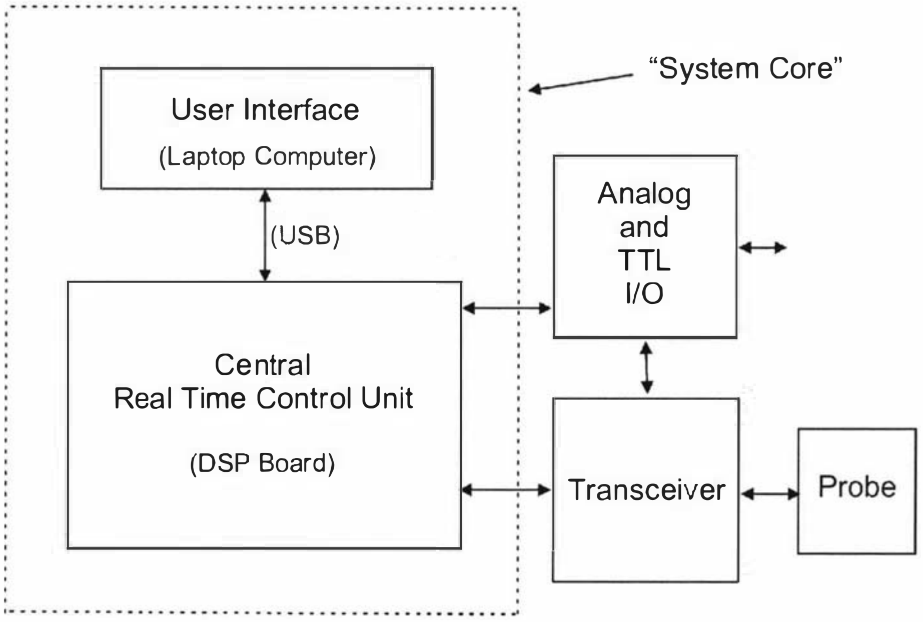


Figure 5.0 NMR system block diagram

The transceiver for a NMR system is basically a quadrature AM radio transceiver. There have been a lot of technology developments in this area that have resulted in many electronic components and techniques becoming readily available for designers. However, with the present trend towards digital cellular radio and UHF/microwave links some of the older and more familiar radio techniques and components are fast becoming obsolete. The block diagram of a NMR transceiver is shown in figure 5.1 and is typical of what exists in a lot of NMR systems today.

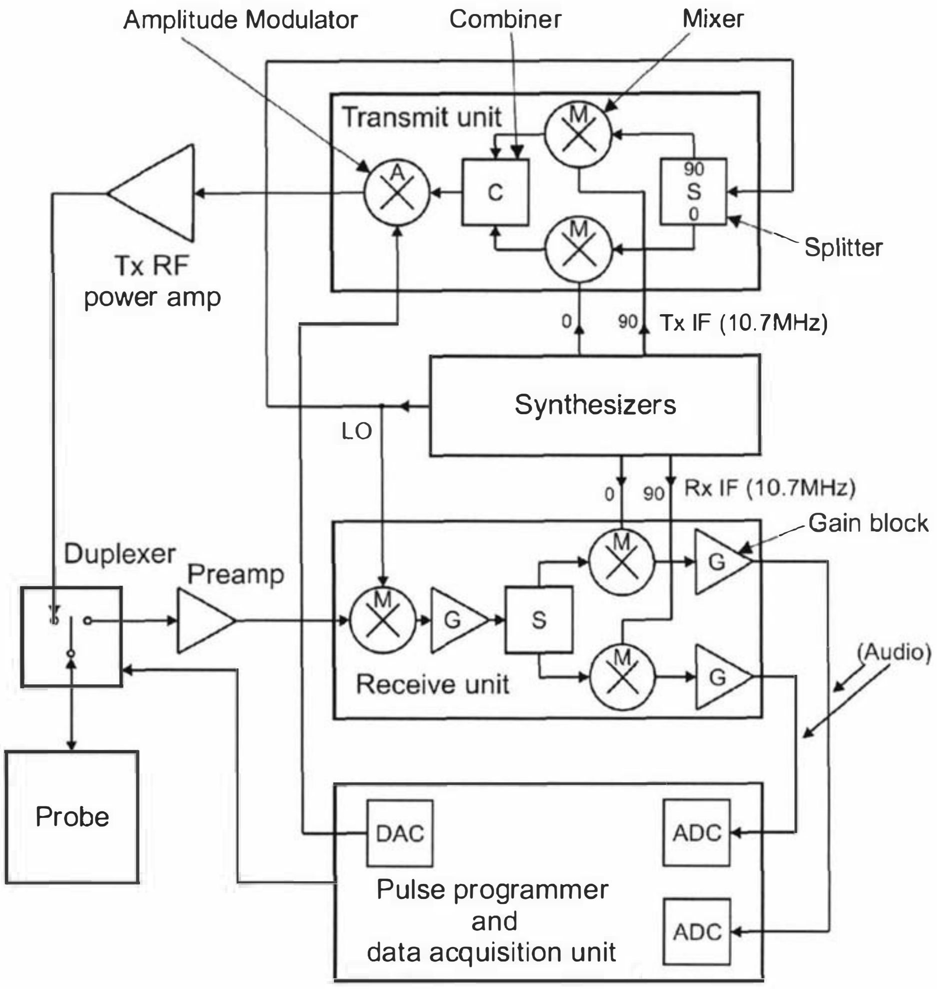


Figure 5.1 Block diagram of a typical superheterodyne based NMR Transceiver.

The transceiver is based on the principles of a superheterodyne receiver whereby intermediate frequencies (IF) are used within the stages during the conversions between radio frequency and audio. The transceiver can be split into three parts, the transmit section, the receive section and the frequency references section. For a single IF stage transceiver, the frequency reference section generates a pair of quadrature phase signals at the IF frequency and another signal at a frequency that is the sum of the IF frequency and the resonant frequency of the nuclei under investigation. For example, if we have a sample that resonates at 200 MHz and we use an IF frequency of 10.7 MHz, the other reference frequency would be 210.7 MHz. This reference is called the local oscillator (LO) and must be phase and frequency locked to the IF frequency sources. This is vital as for NMR the phase and frequency information is important.

The transmit section is responsible for generating a variable phase, amplitude modulated signal at the same frequency as the resonant frequency of the nuclei being investigated. This is done by first mixing a pair of synthesized quadrature IF signals with the local oscillator to produce a signal at the resonant frequency. Amplitude modulation is then applied to change the envelope of the output pulse. The phases of the IF signals can be changed so that the phase of the final output relative to the LO can be changed. This is to provide the phase shifts that are needed during many pulse sequences.

The receive section must process the small but amplified resonance signal from the nuclei and generate a set of base-band quadrature signals that can then be digitised. The input signal (200 MHz) is first mixed with the LO (210.7 MHz) to shift the signal to the IF frequency (10.7 MHz). Then the signal is further amplified and mixed with a set of quadrature IF signals to produce the final base-band signals which are then amplified and low-pass filtered before being digitised. Often the IF frequency of 10.7 MHz is used as this is a common IF frequency for a lot of radio communications equipment and therefore a standardised range of components for 10.7 MHz is available that includes high gain narrow band amplifiers and band-pass filters.

Figure 5.2 (below) shows in more detail the latter part of the receive side, where the conversion between the IF signal and digitised base-band signals take place. This section is often referred to as a phase sensitive detector or quadrature receiver and is a very important part of the NMR system. However it suffers from many problems such as gain mismatch of the amplifiers, filters and mixers as well as DC offsets, phase mismatching between the base-band channels, nonlinearities and poor dynamic range. Gain mismatch and DC offsets are usually reduced by repeating experiments and implementing phase cycling and signal averaging [1].

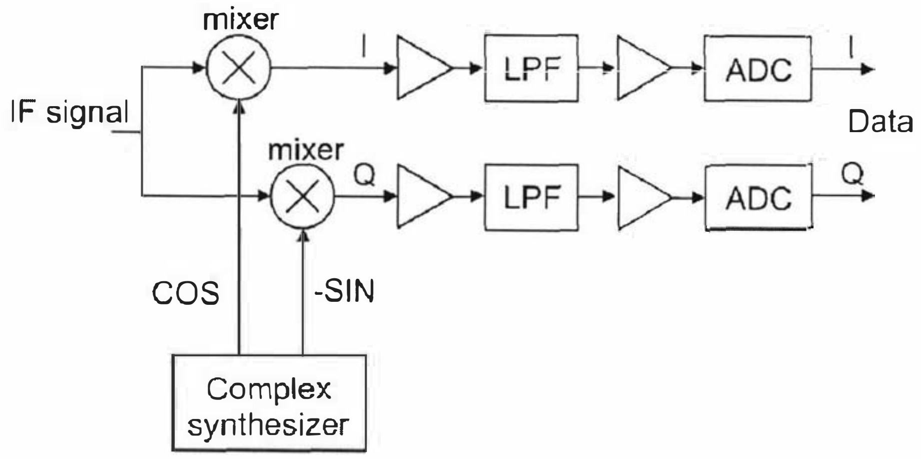


Figure 5.2 The receiver section of the NMR transceiver.

Nowadays, ADCs with 14-bit resolution and sampling rates of 80 Msamples/s (MSPS) are readily available [88]. Therefore we can place the ADC directly at the output of the IF stage and avoid many of the problems associated with the traditional receiver design. This is shown in figure 5.3 where the signal processing is now done digitally. This architecture is therefore referred to as a digital receiver.

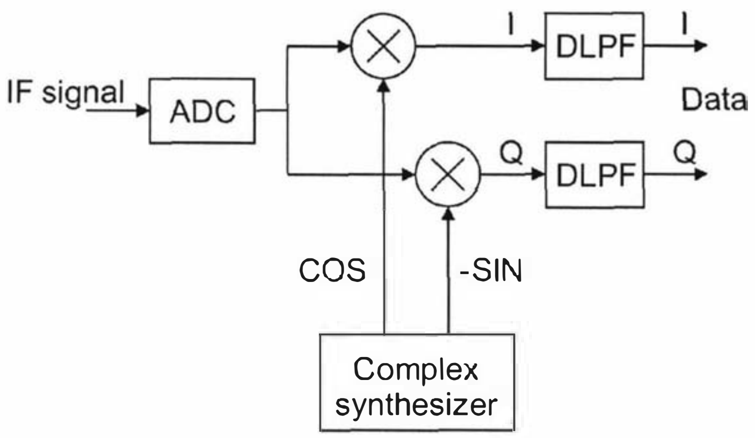


Figure 5.3 Block diagram of a digital receiver

The signal path is very similar to the traditional design where the IF input signal is mixed with the quadrature IF reference signals to produce the base-band signals which are then low pass filtered. The difference is in how it is implemented. First the IF analogue input signal is digitised by a high speed ADC. Secondly, the quadrature IF signals are produced by a numerically controlled oscillator and are digital signals at the same sample rate of the ADC. The digitised input signal is then mixed with the quadrature IF signals using digital multipliers. The result is that the spectral component of the signal is now shifted to base-band but the sample rate is still at the rate of the ADC. This high sample rate is now unnecessary, as a reduced sample rate can be used to preserve the base-band signal information and therefore sample rate reduction and low-pass filtering is performed using Decimating Digital Low-Pass Filters (DLPF).

Low-pass filtering must be used to reduce the bandwidth of the channel before decimation in order to prevent aliasing. This method results in a reduction of the quantisation noise caused by the finite resolution of the ADC. The quantisation noise power is initially spread evenly over the entire input sample rate bandwidth. When the bandwidth is reduced due to low-pass filtering and decimation, the total noise power is also reduced and therefore we have an effective increase in the resolution of the quantised signal. This is referred to as *signal processing gain*. Another way of thinking about this is that the quantised outputs of the ADC are put through a moving average filter and therefore one would expect a reduction of the uncertainty caused by quantisation. This technique is now widely used and forms the basis of Sigma-Delta ADCs where a single bit quantiser is used and the input signal is massively over-sampled.

Another alternative to the traditional analogue receiver that is not so demanding in terms of ADC performance is to have a second lower frequency IF stage and then do the final quadrature detection digitally. This approach has been used by a few NMR system developers and is an intermediate step while high speed A-D and DSP technology is further developed. A suitable lower frequency IF is 455 kHz as it is another standard used in communications. However, today it is not so critical to work with standard IF frequencies as there are now many amplifiers and other devices that can operate over a broad range of frequencies, such as the AD8370 variable gain amplifier that can operate from low frequencies to 750 MHz [89].

As most portable NMR systems will be based on permanent magnet probes, they will typically be operating below about 30 MHz and therefore we can avoid the first mixer and detect the signal directly after the input amplifier stages. This is a big advantage as mixers suffer from nonlinearities and poor dynamic range. It also results in a significant reduction in the number of components and circuit board real-estate. This technique is optimal in terms of getting the digitisation part as close as possible in the signal processing chain to the signal source.

Direct Digital Synthesizer (DDS) devices that are capable of generating signals beyond 100 MHz are now available. By using one of these the transmit section can be greatly simplified. Some of these devices also include a digitally controlled variable output amplitude feature for implementing amplitude modulation. Figure 5.4 shows the block diagram of a complete digital transceiver for my portable NMR systems. This forms the transceiver block that was shown back in figure 5.1.

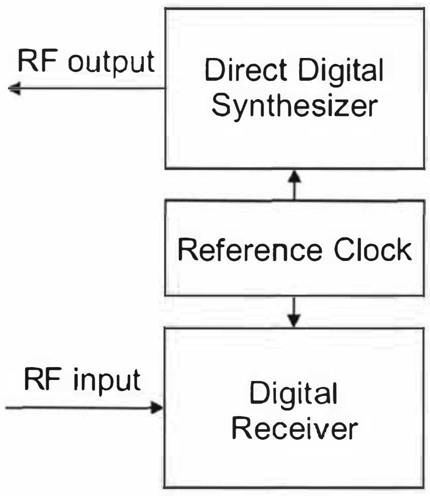


Figure 5.4 Block diagram of a modern digital transceiver

5.2 Digital transceiver hardware design

5.2.1 Transceiver design criteria

After evaluating various existing NMR systems, permanent magnet probe requirements and present electronic technologies, a list of basic design criteria for the transceiver was defined and is summarised as follows:

1. An input frequency range of 0 to 30 MHz to cover the range of most permanent magnet based probes. No mixing is allowed and therefore the A-D conversion process must preserve the desired input bandwidth.
2. Quadrature detection so that phase information can be obtained.
3. A receiver output bandwidth of 2 MHz for each channel. This is to allow for the very short spin-echoes that come from inhomogeneous probes such as the NMR Mouse.
4. Phase locked transmit and receive sections synchronised to the pulse program to allow the exact repeating of experiments and therefore signal averaging.
5. A synchronised 0 to 30 MHz frequency source with rapid phase, frequency and amplitude change capabilities. The variable RF output amplitude is required to set different transmit powers for 90/180° pulses for different probes and for tuning/matching. The RF output should also be gated to minimise noise going to the RF transmit power amplifier during receive.

5.2.2 Transceiver devices.

A number of electronic devices were investigated and the ones that were thought to be the most suitable are now briefly described. When prototyping, some of the biggest problems are obtaining the devices and being able to solder them onto a board. This often constrains the range of devices that can be considered. For the transceiver there was a tendency to use devices from Analog Devices [90] because this company has a wide range of suitable devices that are readily available. They also, like Motorola, provide excellent documentation and design tools. Sticking to one supplier can also make the design process easier as often the devices have been designed to interface directly to each other.

AD6620 Digital Receive Processor

The digital receive signal processor forms the heart of the digital receiver section and today there are a number of these devices that can be obtained from various manufacturers. The devices that were considered were the HSP50016 and HSP50214 from Intersil [91], the GC1011A, GC4014 and GC1012 from Graychip (now owned by Texas Instruments) [92] and the AD6620 and AD6624 from Analog devices. The HSP50016, HSP50214, GC1011A, GC4014 and AD6624 were immediately eliminated as they could not provide the desired output bandwidth. This left the GC1012 and the AD6620. The GC1012 has only 12-bit input, has limited decimation rates, is mounted in a large 120 pin package and is difficult to obtain. Therefore, the preferred device was the AD6620 from Analog devices [93] as it met all the criteria. It was most flexible and efficient in terms of its functionality, is mounted in an easy to solder surface mount package and was easy to obtain.

The AD6620 was developed for digital communications and is made up of four pipelined signal processing elements consisting of a frequency translator, two Cascaded Integrator Comb (CIC) filters and a Finite Impulse Response (FIR) filter. A block diagram of the device is shown in Figure 5.5

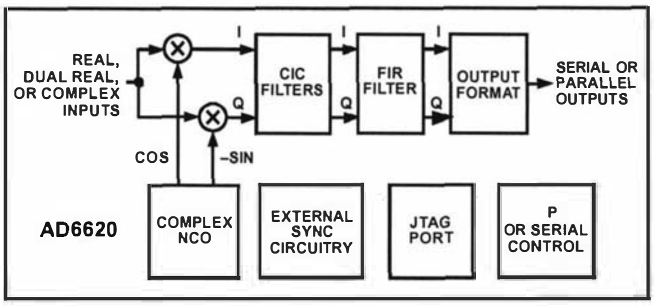


Figure 5.5 AD6620 Functional block diagram. Figure taken from reference 93.

The input data stream from the ADC is first fed into the frequency translator section which mixes the input signal with synthesized sine and cosine functions to shift the signal spectrum to base-band and separate it into quadrature components. This process is all done digitally using two digital multipliers. A Numerically Controlled Oscillator (NCO) generates a set of quadrature local oscillator signals that are digital and at the same rate as the input data sample rate. The output sample rate from the frequency translator is at the input sample rate and is therefore much higher than is required to preserve the now base-band signal. The rest of the AD6620 is concerned with reducing this excessive sample rate and is done using three pipelined decimating filters.

Decimating fixed coefficient CIC filters or Hogenauer filters [94, 95] are low-pass multi-rate filters that are used to implement large sample rate changes in digital systems. They are basically a unity coefficient moving average filter with a sin(x)/x low-pass response and are often referred to as a *sinc* filter. Since the filter uses unity coefficients, no multipliers are needed and so only adders and subtracters are required. This makes them simple and efficient in terms of functionality, performance and silicon real-estate. The characteristics of the filter are mainly determined by the decimation rate chosen. To improve the CIC filter anti-aliasing and stop-band attenuation, multiple sections are cascaded to increase the order of the filter. Since CIC filters have a sin(x)/x like pass band response, conventional FIR filters are often used to compensate for this. The general idea is to use CIC filters to do the transition between high and low sample rates and then use a FIR filter at the low sample rate to clean it up. This is a very efficient method and has been adopted by many producers of digital receivers.

The first decimating section within the AD6620 is a 2^{nd} order CIC filter (CIC2) that has, for a decimation rate of 4, the frequency response shown below in figure 5.6. This section is the first stage in the chain. It operates at the input sample rate and is capable of decimation rates from 1 to 16. It determines the operating rate and consequently the power consumption of the following sections. To minimise the power consumption of the device one would aim for the highest possible decimation rate, but this leads to reduced rejection of aliased components and is therefore not ideal. So there is a trade off between power consumption and performance.

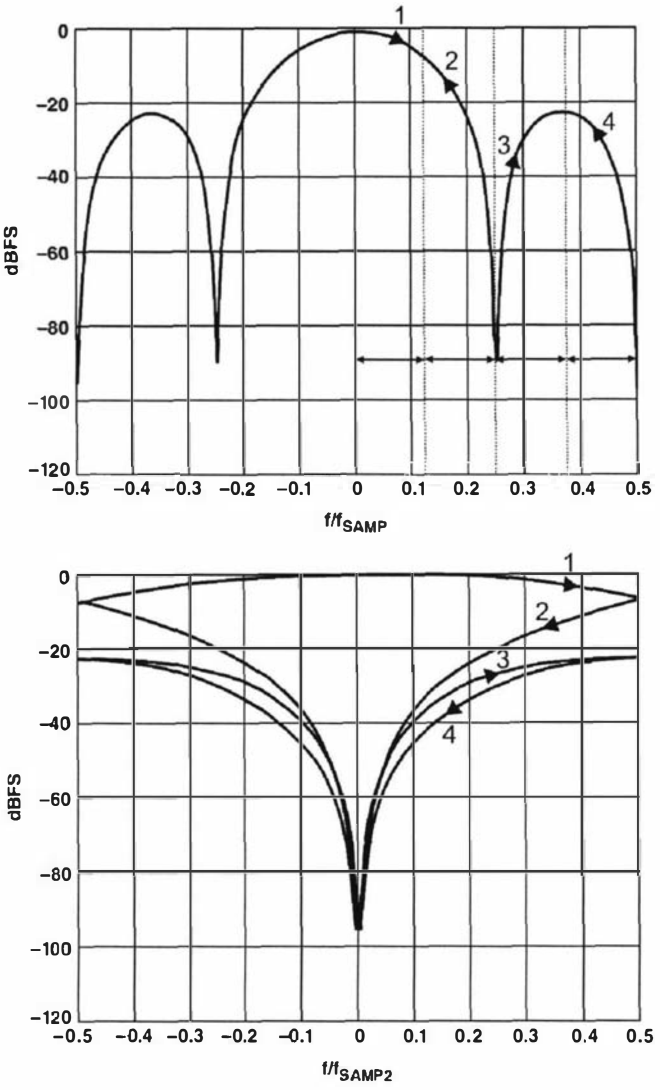


Figure 5.6 AD6620 CIC2 filter with decimation rate of 4. f_{SAMP} is the input sample rate and f_{SAMP2} is the output sample rate. Figure taken from reference 93.

The upper plot of figure 5.6 is referenced to the input sample rate and shows the frequency response of the filter. When the sample rate is reduced, any frequency components beyond the output sample rate Nyquist frequency are repeatedly folded back in concertina fashion. In the present example, the decimation rate of 4 resulted in the Nyquist frequency with respect to the input sample rate being 0.125. The frequency components that are within this Nyquist range (section 1) are fully preserved, but the other frequency components (sections 2-4) get aliased. The lower plot is referenced to the output sample rate and shows this folding process. Fortunately these aliased components are reduced in amplitude by the low-pass response of the filter and are also minimal at the centre of the desired pass-band. However, one should always try to reduce the aliasing in order to improve the signal to noise performance of the receiver. This can be achieved by minimising the signal bandwidth with respect to the output sample rate, in other words keep the signal bandwidth within section 1. This can be done by either minimising the decimation rate used, which maximises the spectral width of section 1, or using channel shaping to minimise the signal bandwidth, such as a band-pass filter on the input to the ADC.

Another set of plots for the CIC2 filter but this time with a decimation rate of 16 is shown in figure 5.7 and shows how the number of filter lobes and aliased components increase. This high decimation rate would therefore only be suitable for very narrow band signals.

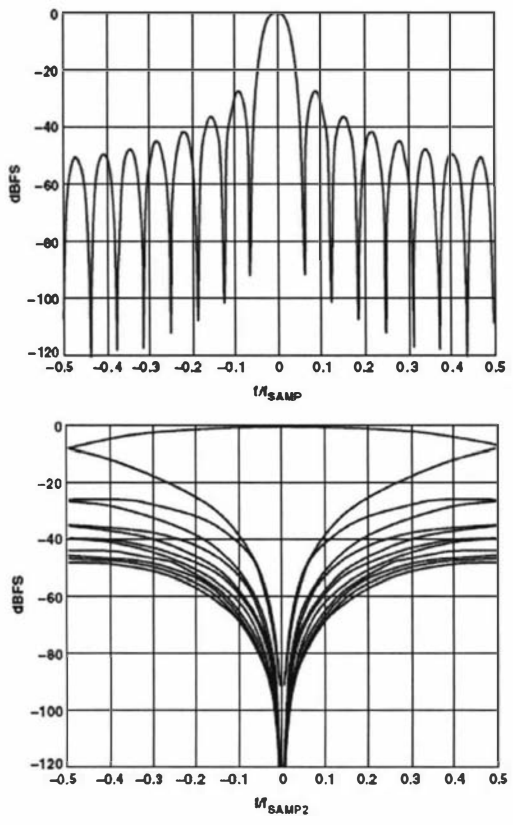


Figure 5.7 AD6620 CIC2 filter with decimation rate of 16. Figure taken from reference 93.

The second decimating section is a 5th order CIC filter that provides decimation rates between 1 and 32 and its frequency response and alias rejection plots for a decimation rate of 4 are shown below in figure 5.8. Here you can see the much improved response and alias rejection performance due to the higher order filter. Figure 5.9 shows the performance of this filter for a decimation rate of 32 and again this rate would be used for a sufficiently narrow band signal. Normally most of the decimation within the AD6620 would be performed within the CIC5 filter, but for very narrow band signals the higher CIC2 decimation rates can be used to reduce device power consumption.

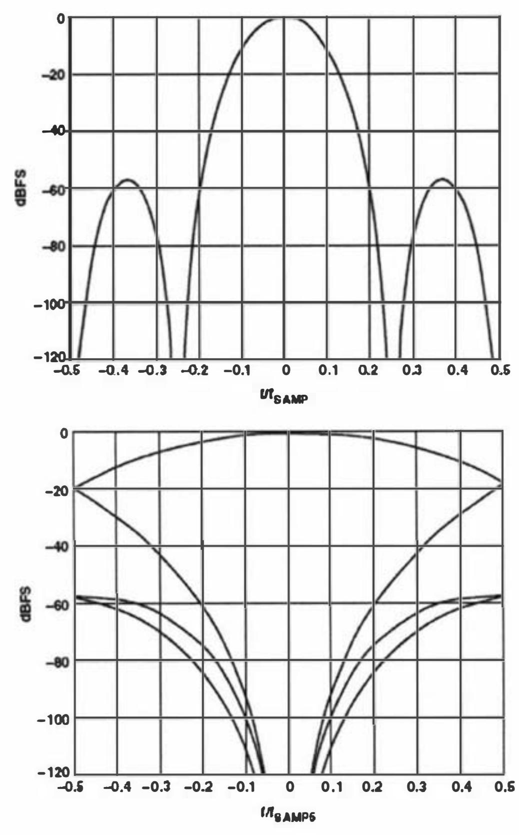


Figure 5.8 AD6620 CIC5 filter with decimation rate of 4. Figure taken from reference 93.

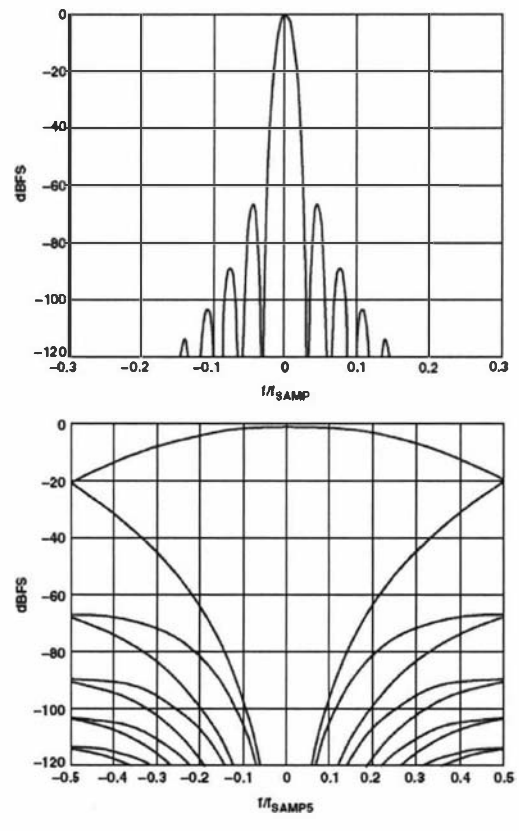
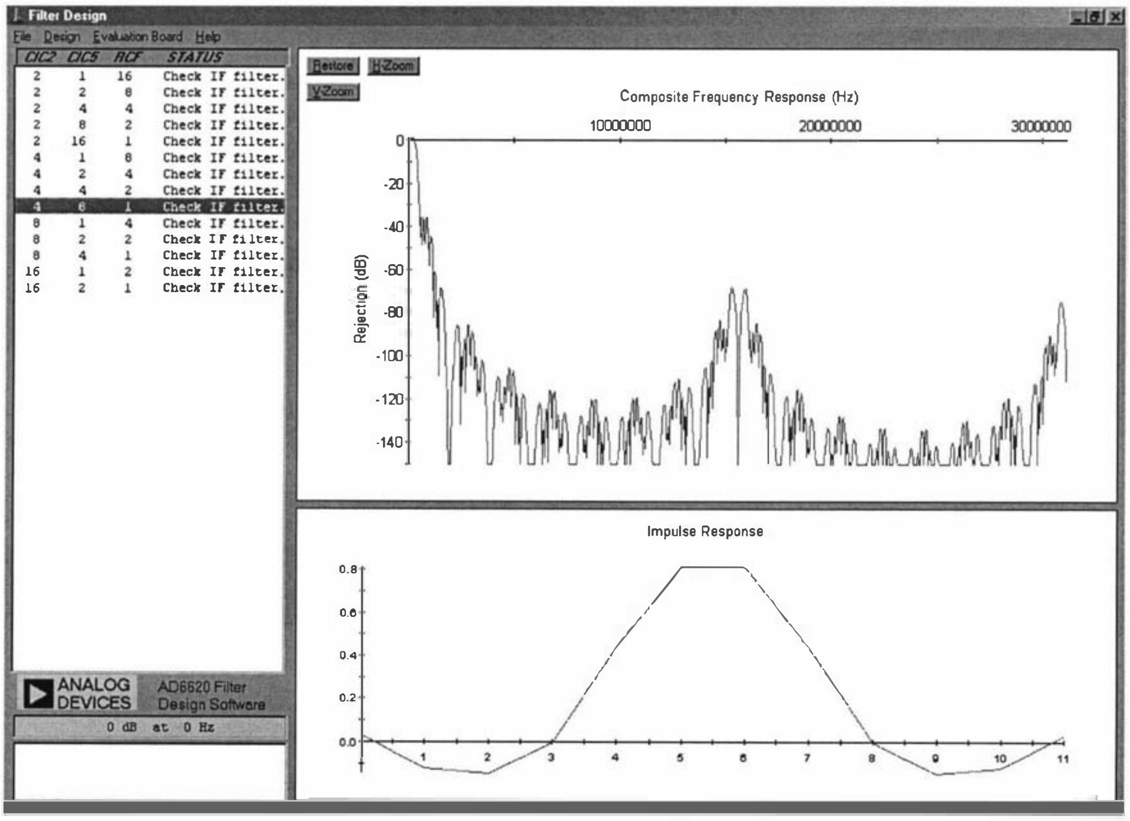


Figure 5.9 AD6620 CIC5 filter with decimation rate of 32. Figure taken from reference 93.

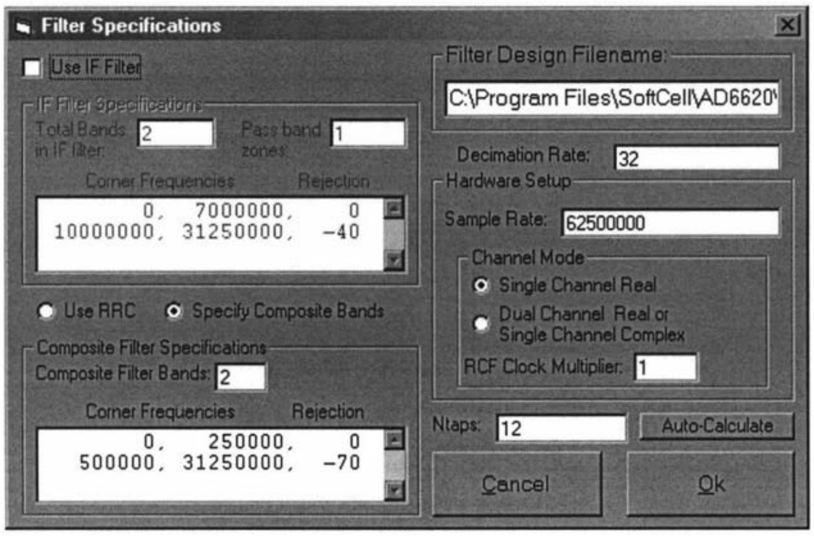
The last decimating section is the Ram Coefficient Filter (RCF). It is a standard FIR filter section that can also provide decimation from 1 to 32. With this section the user determines the filter coefficients and therefore has control over the frequency response. Up to 256 18-bit coefficients can be used, but this is determined by the desired output bandwidth and the clock rate of the device, which is usually the input data rate. The AD6620 can only calculate one filter tap per clock cycle and so if the output data rate is very high then the device does not have enough time to use many coefficients. If decimation is used within this section then just like the other decimating sections one needs to consider the rejection of aliased components. The FIR section enables sharp edged filters to be implemented and it is also often simultaneously used to compensate for the poor pass-band performance of the CIC filters.

Filter design software is provided by Analog Devices to simplify and assist with choosing the various decimation rates for each section and also the generating of the FIR filter coefficients. The user defines a desired response and total decimation rate and then the software calculates a series of optimal filter responses for different combinations of section decimations rates. There is also the option of having the channel bandwidth specifications included within the total system response calculations. The user can view the various responses and when satisfied they can generate a FIR coefficient file. Figures 5.10 and 5.11 show the user interface to the filter design tool.



CIC2	CIC5	RCF	STATUS
2	1	16	Check IF filter.
2	2	8	Check IF filter.
2	4	4	Check IF filter.
2	8	2	Check IF filter.
2	16	1	Check IF filter.
4	1	8	Check IF filter.
4	2	4	Check IF filter.
4	4	2	Check IF filter.
4	8	1	Check IF filter.
8	1	4	Check IF filter.
8	2	2	Check IF filter.
8	4	1	Check IF filter.
16	1	2	Check IF filter.
16	2	1	Check IF filter.

Figure 5.10 AD6620 filter design software response plots [96]. The frequency response (upper plot) is shown relative to the input sample rate and is the combined response of the CIC and FIR filters. The lower plot is the impulse response of the FIR filter. In this case 12 coefficients are used with weightings between +1 and -1. The table on the left hand side shows the possible decimation combinations. The chosen one is highlighted.


Filter Specifications

☐ Use IF Filter

IF Filter Specifications

Total Bands in IF filter: 2 Pass band zones: 1

Corner Frequencies Rejection

```
          0,  7000000,     0
   10000000, 31250000,   -40
```

○ Use RRC ● Specify Composite Bands

Composite Filter Specifications

Composite Filter Bands: 2

Corner Frequencies Rejection

```
          0,   250000,     0
     500000, 31250000,   -70
```

Filter Design Filename: C:\Program Files\SoftCell\AD6620\

Decimation Rate: 32

Hardware Setup

Sample Rate: 62500000

Channel Mode

● Single Channel Real

○ Dual Channel Real or Single Channel Complex

RCF Clock Multiplier: 1

Ntaps: 12

Figure 5.11 AD6620 filter design software specification panel [96].

The AD6620 allows the outputs of the various filter stages to be scaled to avoid overflow and also provides the option to bypass stages. A detailed block diagram that also includes interfacing details is shown in figure 5.12. The data from the AD6620 can obtained through either a serial or parallel interface. The serial interface is provided for narrowband applications and reduces the number of data lines from 16 to 1. Control of the device is performed through an 8-bit microprocessor bus compatible interface and synchronisation input pins are provided so that each of the sections can be synchronised to an external signal. The AD6620 comes in a 80-pin surface mount package and operates from a single 3.3 V supply.

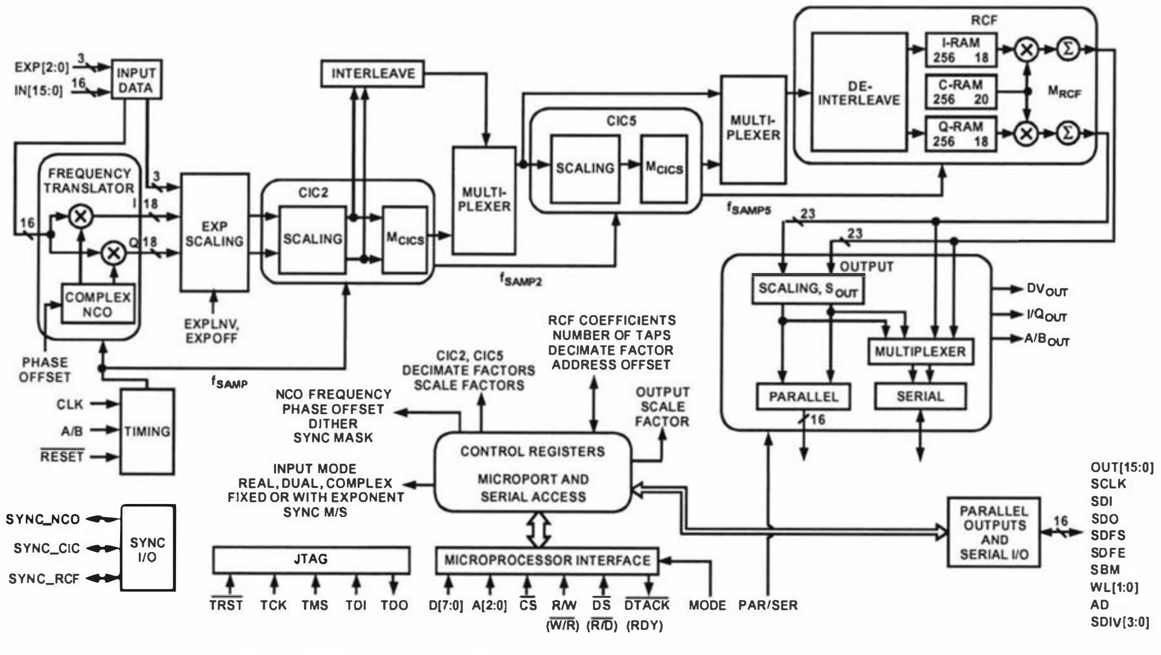


Figure 5.12 AD6620 detailed block diagram. Figure taken from reference 93.

AD6644 A-D Converter

The Analog Devices AD6644 [88] was chosen as it can sample up to 65 MSPS with 14-bit resolution and can be directly connected to the AD6620. This is a three stage converter where multiple lower resolution A-D sections combined with track and hold and DACs are used in a pipeline fashion to do the conversion process. A block diagram showing the arrangement of these sections is shown below in figure 5.13. This technique results in reduced power consumption and smaller device die sizes.

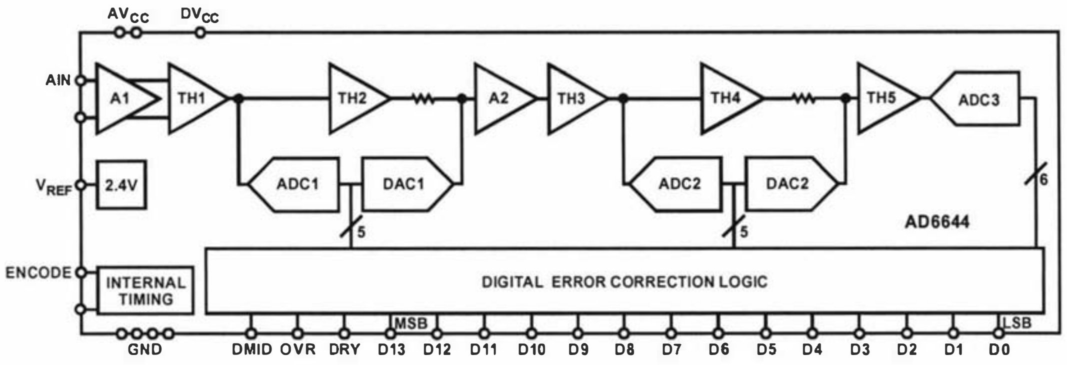


Figure 5.13 Block diagram of AD6644 ADC internal architecture. Figure taken from reference 88.

The analogue signal and encode clock inputs are both differential inputs. This is common for high speed, high dynamic range converters and results in improved performance as differential inputs have high common mode rejection to unwanted signals such as supply and ground noise. Each analogue input has a 250 MHz bandwidth and a maximum input swing of ±0.55 V centred around a 2.4 V reference. This allows for a differential input signal of up to 2.2 Vpk-pk. The 2.4 V reference voltage is provided on an output pin for the correct biasing of an external driver. 5 V and 3.3 V power supplies are required and the device typically consumes 1.3 W so that some heat-sinking is necessary.

AD8131 Differential Driver Amplifier

The Analog Devices AD8131 [97] is a differential driver amplifier that can be used to generate the necessary differential signals for the AD6644. Figure 5.14 shows the pin arrangement and internal functionality of the device. It has a fixed gain of two and a full power bandwidth of 400 MHz. A common mode input pin is provided so that the output can be biased to suit the requirements of a differential input ADC.

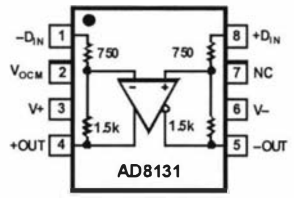


Figure 5.14 AD8131 differential driver functional diagram. Figure taken from reference 97.

As the input amplifier and the ADC have a wide analogue bandwidth, sub-sampling could be used. A sample rate of at least twice the bandwidth is required to preserve the information. Sub-sampling is a simultaneous mixing and sampling process. The signal is shifted in the frequency domain by the difference between its centre frequency and the sampling rate. However there is a drawback since it requires a very low phase noise oscillator and conversion process. Any phase jitter can cause large distortions. The approach at this stage was to play safe and make sure that the sample rate was always at least twice the highest frequency component of the signal. This allows the use of a standard oscillator module.

AD9852 Direct Digital Synthesiser

The AD9852 [98] is a high performance Direct Digital Synthesiser (DDS) that is one of a series of DDS devices available from Analog Devices. A block diagram of the architecture is shown below in figure 5.15. The device uses a 12-Bit DAC and is capable of an output update rate of 300 MHz. The synthesised output signal frequency, phase and amplitude can be rapidly altered and are each set using registers that are programmed through either a parallel 8-bit microprocessor compatible bus or a serial bus. On-chip clock multipliers are included so that a range of external clocks can be used and a synchronisation input pin is provided so that synchronisation with other devices can be achieved.

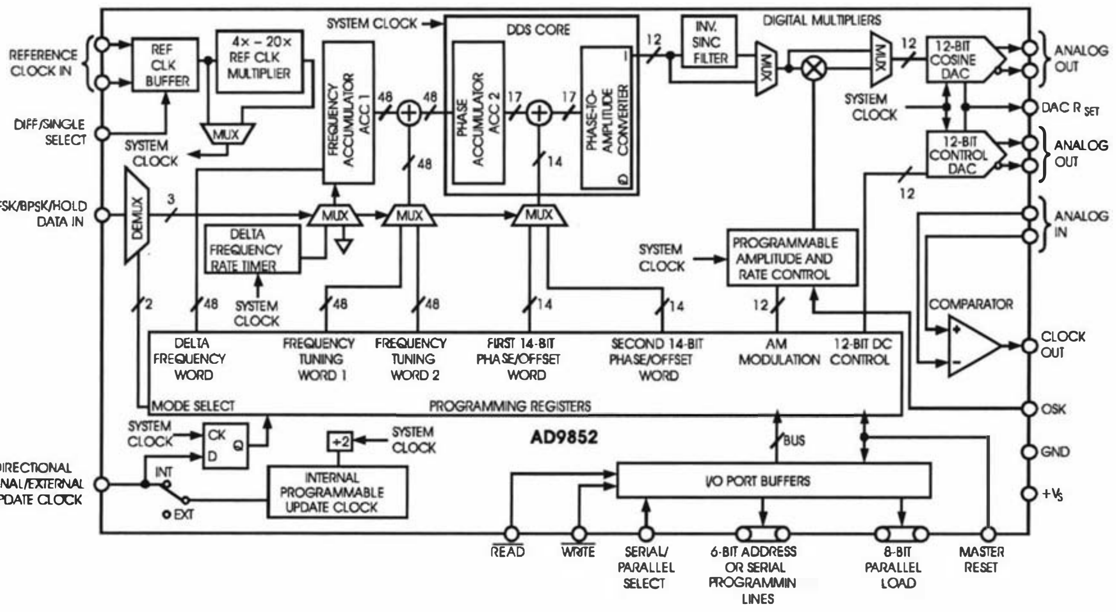


Figure 5.15 AD9852 block diagram. Figure taken from reference 98.

ADG901 RF Switch

The ADG901 [99] is a new type of RF switch from Analog Devices and was chosen as it provided very good off isolation and virtually no charge injection during switching. These two characteristics are important as the device is used to isolate an RF signal from the input of a high power RF amplifier. Other devices used in the past have injected a considerable amount of charge into the circuit which resulted in a voltage spike being fed to the input of the RF amplifier, leading to a large unwanted output transient. An internal schematic of the device is shown in figure 5.16 and highlights that it is designed for 50 Ω RF systems.

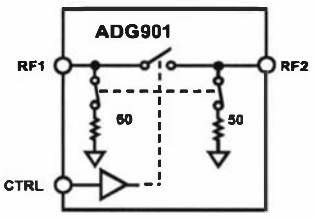


Figure 5.16 ADG901 internal schematic. Figure taken from reference 99.

5.2.3 Transceiver circuit design and layout.

After evaluating the preferred devices, the design criteria and also other requirements, such as the DSP clock, a transceiver system block diagram was drawn up and is shown below in figure 5.17. This then became the basis for the detailed circuit design.

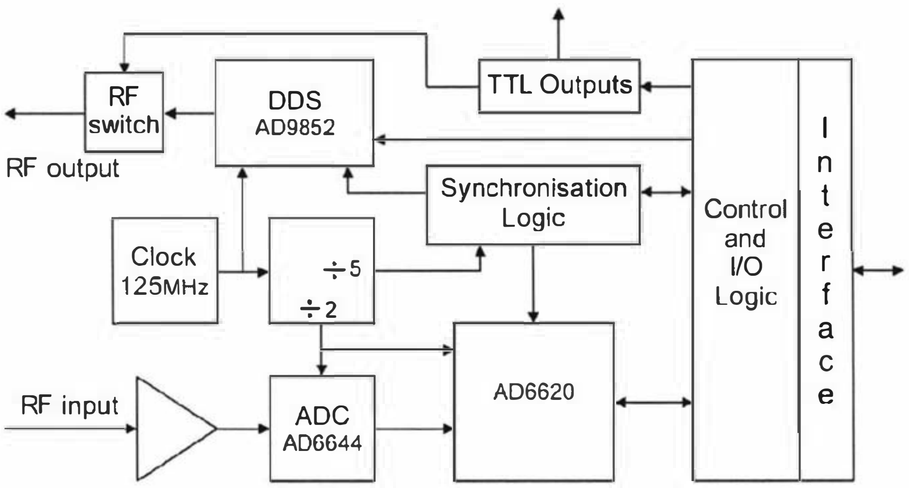


Figure 5.17 Transceiver detailed block diagram

An important part of the design was determining the device clock rates and how to synchronise the AD6620, the AD9852 and the DSP. A greater than 25 MHz reference clock was needed for the DSP to meet the requirement that, in order to maintain a constant phase relationship between the DSP reference clock and the DSP system clock, the clock multiplication factor has to be less than 5. The next requirement was to operate the AD6620 as close as possible to its maximum clocking rate of 65 MHz. This was achieved with a master 125 MHz reference clock divided by two to get 62.5 MHz for the AD6620, divided by five to get the 25 MHz clock for the DSP and used directly for the reference clock for the AD9852 DDS. The synchronisation logic further divides the 25 MHz clock by two to generate a common "heartbeat" clock that signals when all the clocks have a transition at the same time. This is illustrated below in figure 5.18 where the rising edge of the 12.5 MHz clock is aligned with the rising edges of all the other clocks.

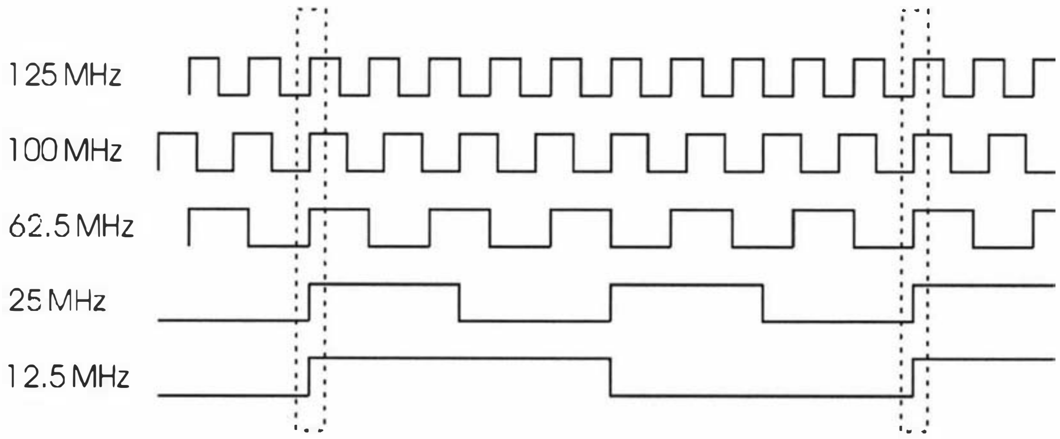


Figure 5.18 Clock synchronisation

The circuit used to generate all the clocks is shown in figure 5.19. Emitter Coupled Logic (ECL) was used as it can operate well above 125 MHz, which is near the upper limit for most other logic device technologies. It has typical propagation delays of less than 1ns resulting in minimal phase noise and skew between the various clocks. ECL also provides differential outputs which are necessary to drive the AD6644 ADC and AD9852 DDS.

The circuit starts with a 125 MHz oscillator module (XTAL1) that has a LVTTL output. This output is then converted to ECL using an MC100EPT20D logic level translator [100] (U4) device and is then passed on to the AD9852 DDS and clock dividers. The 62.5 MHz and 25 MHz clocks are generated using an MC100EPT139DT programmable clock divider [101] (U20) that can be set to provide outputs with divisors of two or five. A 90 degree phase shifted (quadrature) version of the 62.5 MHz clock is generated using an MC100EP52D D flip-flop [102] (U8). This clock is produced by feeding the main 62.5 MHz clock into the D input of the flip-flop and using an inverted version of the 125 MHz clock as the clock input. Inversion is done by swapping the differential ECL inputs. The last output is the 12.5 MHz heartbeat clock and again it is produced using an MC100EP52D D flip-flop (U13). This time the quadrature version of the 62.5 MHz clock is fed into the D input and the 25 MHz clock is used to clock the flip-flop. The two 62.5 MHz clocks as well as the 25 MHz and 12.5 MHz clocks are then converted to LVTTL levels using MC100EPT23D translator devices [103] (U6 and U21).

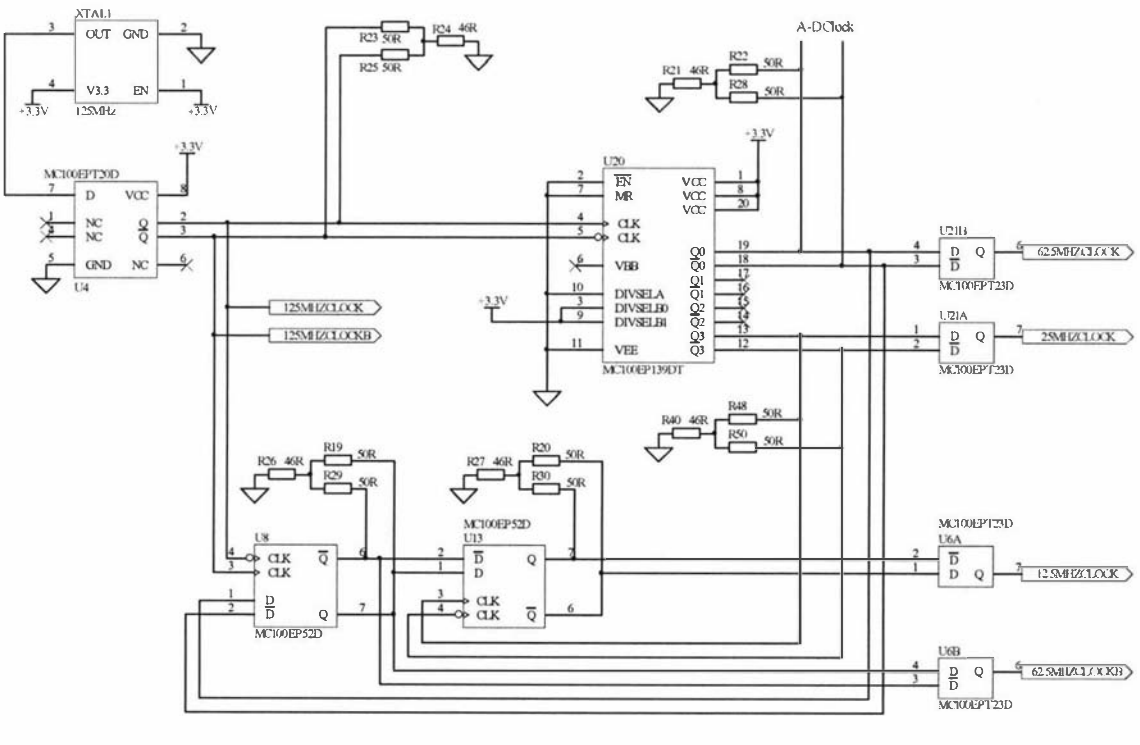


Figure 5.19 Clock generation circuit

The 12.5 MHz heartbeat clock is then used to synchronise the AD9852, the AD6620 and the DSP pulse program. The DSP will always have its clock synchronised, but there is no guarantee that the pulse program will always start execution at the same time relative to the common 12.5 MHz clock and so a method is needed to get the DSP pulse program synchronised. This is done using some logic and a software loop within the pulse program that repeatedly writes a byte to the I/O port until it detects that a DSP write cycle has lined up with one of the 12.5 MHz clock edges. The delay between subsequent writes is an odd multiple of 10 ns. Therefore after a few write cycles synchronisation should be obtained. The logic that is used to implement this feature is shown in figure 5.20. Figure 5.21 shows a write cycle timing diagram and the generation of a pulse that is applied to the D input of the flip-flop (U19B). When the pulse is aligned with the rising edge of the 12.5 MHz clock the output of the flip-flop goes high, signalling to the DSP via a timer input pin that synchronisation has been detected.

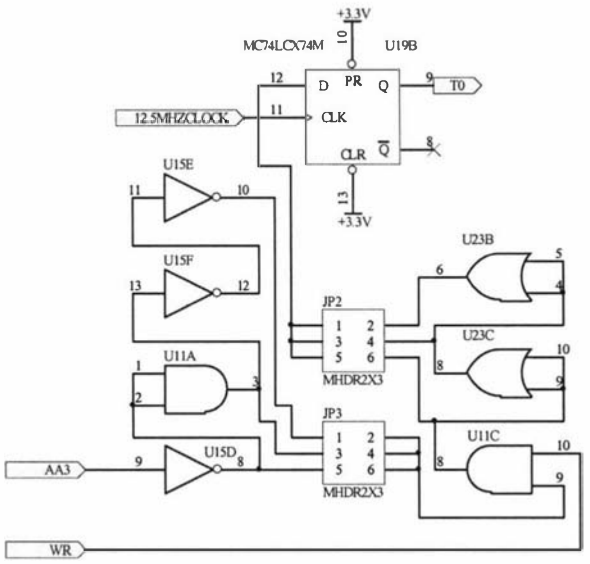


Figure 5.20 DSP synchronisation circuit. JP2 and JP3 are jumper blocks with the default configurations for both being connections between pins 5 and 6.

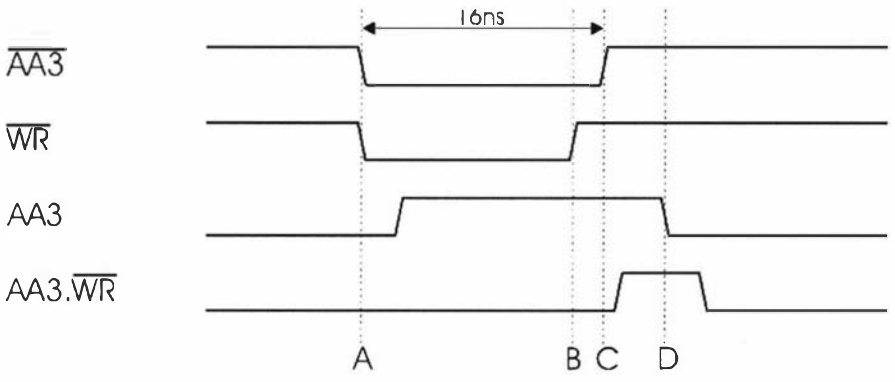


Figure 5.21 DSP write cycle timing and pulse generation.

The write cycle is always referenced to the 100 MHz DSP clock and with 1 wait state it takes about 16 ns. The write cycle starts with $\overline{AA3}$ (AA3 in schematic) and $\overline{WR}$ (WR in schematic) both going low (A) and finishes with $\overline{WR}$ going high (B) followed by about 2 ns later by $\overline{AA3}$ going high (C). Therefore there is a short time when $\overline{AA3}$ is low and $\overline{WR}$ is high (B-C). This is used to generate a narrow pulse for synchronising with the 12.5 MHz clock. This is done by firstly inverting $\overline{AA3}$

(using U15D) and then ANDing it with $\overline{WR}$ (using U11C) to produce $AA3\cdot\overline{WR}$. The logic used to invert $\overline{AA3}$ introduces a delay which makes the overlap with $\overline{WR}$ longer (B-D) so after ANDing results in a longer pulse, which is also delayed by the AND gate. The overlap (B-D), and hence the length of the pulse, can be further increased by inserting additional gate delays which is done by selecting the appropriate jumper connection for JP3. The output from the AND gate can also be further delayed by inserting extra gate delays (JP2) therefore allowing the shifting in time of the pulse relative to the DSP 100 MHz clock. For correct operation the pulses should be less than 10 ns in duration (the timing resolution of the DSP) and with centres aligned with the rising edge of the 12.5 MHz clock. To meet the set-up and hold requirements (t_S>2.5 ns, t_H>1.5 ns) of the MC74LCX74 [104] flip-flop it is important that the pulse is long enough and that there is sufficient time between the rising edge of the pulse and the rising edge of the 12.5 MHz clock. The extra gates were included when designing the circuit to allow for the uncertainties in the propagation delays of the logic devices and the circuit layout. However it was found that after constructing several transceiver boards that no additional gate delays were necessary. Therefore the default condition for both JP2 and JP3 is a connection between pins 5 and 6. The propagation delays for MC74LCX logic gates [105, 106] are specified as being between 1.5 and 5.5 ns for both t_{PHL} and t_{PLH}.

The DSP code that is used for synchronisation is reproduced below as figure 5.22 and a timing diagram showing how synchronisation is achieved is shown in figure 5.23. The write cycle is repeated every 70 ns until a pulse lines up with the rising edge of the 12.5 MHz clock. A timer input is used to detect synchronisation by polling the timer status register after every write cycle to see if a flag is set. Once synchronisation is detected the software loop is exited and the pulse program is then executed.

```
        movep   #$0F3FE1,x:A_BCR    ;Set up wait states, 1 for AA3
        movep   #0,x:A_TLR0         ;Set up synch timer
        movep   #0,x:A_TCSR0        ;Disable timer
        movep   #$200261,x:A_TCSR0  ;Set up timer0 input capture
        move    #$80,a1
synchl  move    a1,x:$20000         ;Start of loop, write cycle
        nop
        jclr    #21,x:A_TCSR0,synchl      ;Synch detected?
        nop
        movep   #$0FFFE1,x:A_BCR    ;Set up wait states, 7 for AA3
        nop
```

Figure 5.22 DSP code for write cycle and pulse generation.

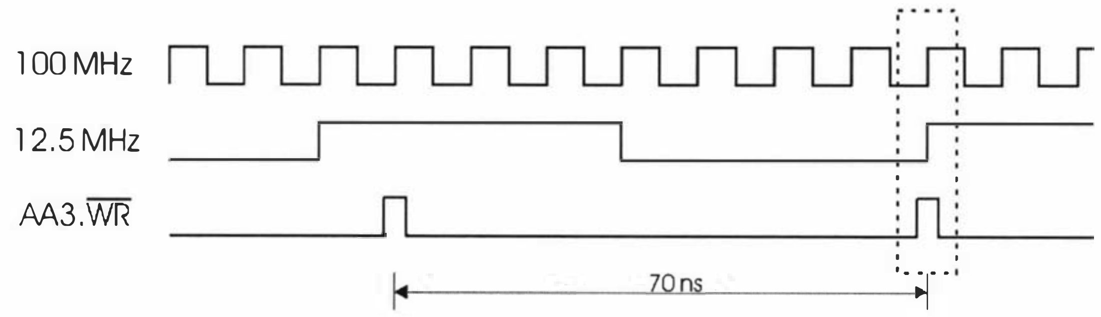


Figure 5.23 DSP synchronisation timing.

The complete schematics for the digital receiver is divided into three sections, the first of these is the clock generation and ADC section and is shown in figure 5.24. This section can be considered the most important as the quality of the analogue to digital conversion process determines the performance of the entire digital receiver. The two main factors that determine the conversion process quality are the number of quantisation levels and the sampling jitter.

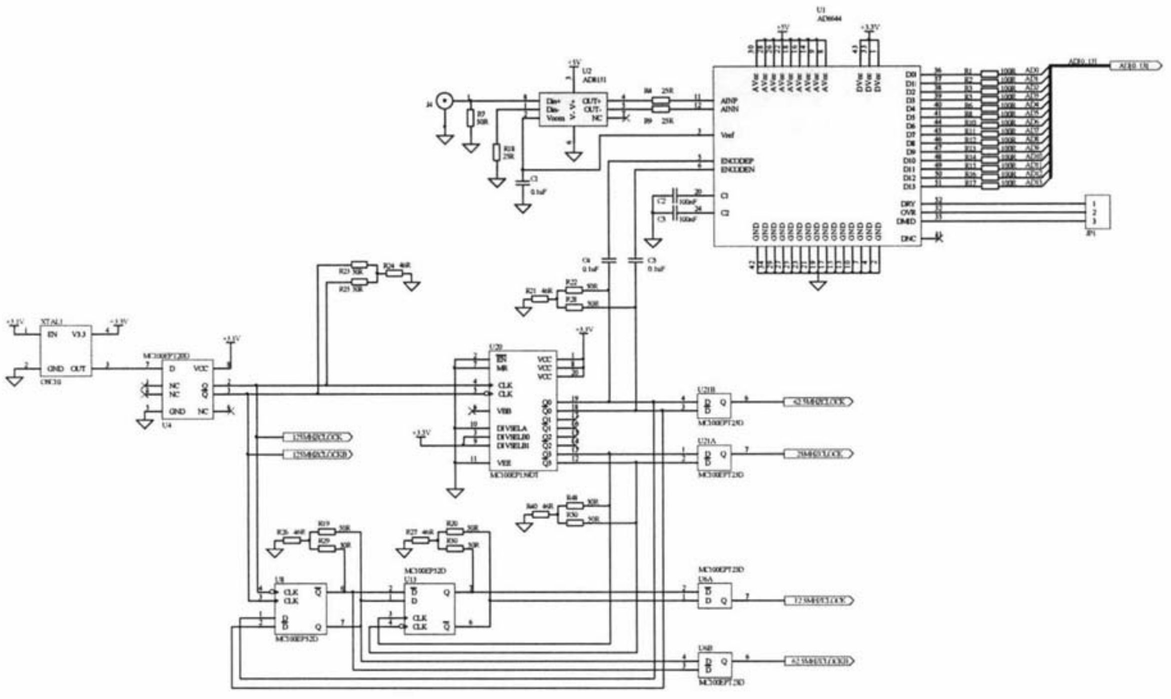


Figure 5.24 Clock generation and A-D section.

The AD6644 is a 14-bit converter and so provides adequate precision and it also boasts a typical sampling jitter of 0.2 ps rms. Any jitter in the sampling time translates to a deviation in the sampled value. The worst case is for an input sine-wave when it is going through the zero crossing as this is when the rate of change is a maximum and for a sine-wave is simply equal to $2\pi f V$ [107] where V is the amplitude. So for a known sampling jitter time (Δt) and a maximum allowable value error (Δv) we can calculate the maximum input frequency using:

$$f = \frac{1}{2\pi}\frac{\Delta v / V}{\Delta t} \tag{5.1}$$

For a 14-bit converter with a 2 Vpk-pk input range ($V = 1$) the value of the least significant bit will be 122 μV. Therefore with a typical sampling jitter of 0.2 ps rms, to maintain 14 bits precision, the maximum allowable input frequency would be about 100 MHz. The sample jitter is also determined by the sample clock and so the maximum allowable input frequency would be further reduced. Therefore a very low phase noise clock should be used for the best performance and is essential if sub-sampling is performed. In this case sub-sampling was not required so a Conner CWX823 crystal oscillator module [108] with a typical jitter of 1 ps rms was used thereby limiting the maximum signal input frequency to 20 MHz. A lot of standard oscillator modules have typical phase jitters of around 20 ps rms and if used would significantly degrade the performance of the receiver. The ECL logic used in the clock circuit has a typical output jitter of 0.2 ps rms. This is the same as for the AD6644. In the future the development of a very low phase noise oscillator may be investigated to allow the receiver to operate with much higher input frequencies and therefore be used with superconducting magnet systems. The receiver can still be operated with higher input frequencies but with reduced signal to noise performance. However, during the decimating process signal to noise performance is improved by reducing the bandwidth of the quantisation noise. The 100 Ω resistors R1 to R17 are provided to reduce the dynamic loading of the AD6644 outputs. A full-scale transition causes all the outputs to change at the same time, creating a large current demand on the output stage of the device. If the dynamic loads on the outputs are too large it could result in an excessive current demand which would lead to a large spike being generated on the power supply. Again this highlights the need to take special care when designing with high speed circuits as not only the loading by other logic devices needs to be considered, but the capacitive loading of the tracks as well.

The analogue input to the receiver is first buffered and converted to two differential signals by the AD8131 amplifier before it is applied to the AD6644. The differential signals are centred around a 2.4 V DC offset that is provided by the AD6644. Resistors R4 and R9, connected in series with the differential outputs of the AD8131, are used to limit the current due to the dynamic load presented by the inputs of the AD6644. This dynamic load results from the input capacitance and protection diodes. The AD6644 has a differential input voltage range of 2.2 V and the AD8131 amplifier has a fixed gain of 2 and therefore the input voltage range of the receiver is 1.1 Vpk-pk. The input impedance of the receiver is 50 Ω.

The next section contains the AD6620 digital receive processor and the AD9852 DDS and is shown in figure 5.25. The AD6620 is clocked at 62.5 MHz and it receives a 14-bit data stream at the same rate from the AD6644. The AD6620 processes this data stream and then outputs *I*nphase and *Q*uadrature data (*I*/*Q*) at the selected decimated rate via its 16-bit output port. When the AD6620 has some data ready it sets its DVout pin high for two clock periods. This is then used to signal the DSP via a timer input pin that new data is available. During the first of these clock periods the *I* value is placed onto the 16-bit data port and the *I*/*Q*out pin is set high. During the second clock period the *I*/*Q*pin is set back to low and the *Q* data is placed onto the data port and will remain there until the next set of data is available. The *I* data only remains on the port for a short time and so latches U7 and U9 are used to capture it. A rising edge trigger signal for the latches is generated by ANDing the *I*/*Q*out signal with the 62.5 MHz clock. The output data, the DVout pin and the *I*/*Q*pin are valid just before the rising edge of the 62.5 MHz clock so ANDing *I*/*Q*out with the 62.5 MHz clock guarantees that the latches will capture the correct data. When the DSP performs a read of the data it sets the output enable pin of the latches low. At the end of the read cycle it sets it high again. This low to high transition triggers flip-flop (U17A) to generate a pulse (using U17B) which then retriggers the latches to store the *Q* data remaining on the data port. The DSP then performs another read to obtain the *Q* data from the latch.

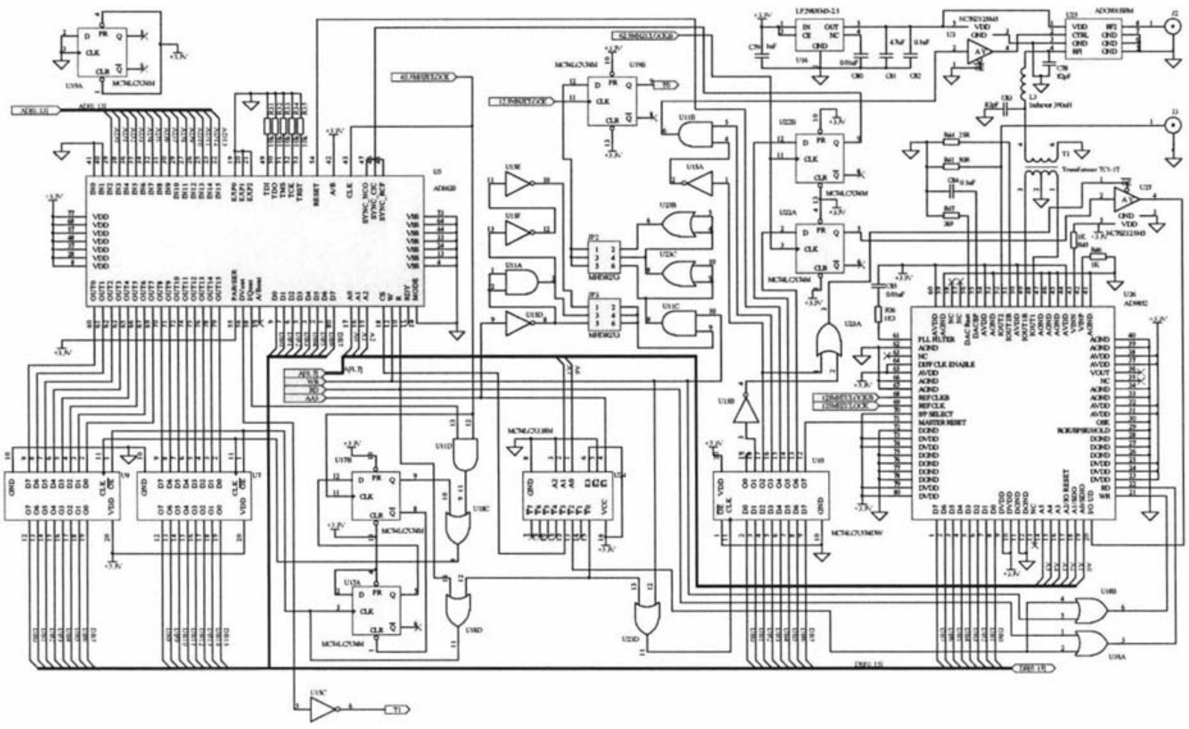


Figure 5.25 Digital receive processor and DDS section (larger version on page 173).

The AD6620 and AD9852 both use 8-bit microprocessor compatible ports for the setting of device parameters. An address decoder (U24) together with some other glue logic is used to map the devices as simple memory locations within the DSP address space. The AD6620 has three input pins that can be used to synchronise its NCO and filters to external triggers. As everything is synchronised to one common reference the three inputs are tied together. The AD9852 can have its NCO started and/or updated synchronously with an external update pulse applied to its I/O-UD pin. The DSP Synchronisation circuit as described earlier is also contained within this section. Once the DSP is synchronised with the 12.5 MHz clock it can then be used to synchronise the AD6620 and the AD9852 by triggering the appropriate input pins.

This is done by first writing some data to the latch U10 and then using the quadrature version of the 62.5 MHz clock and two flip-flops (U22A and U22B) to generate rising edges that are not aligned with the rising edges of either the 125 MHz or 62.5 MHz reference clocks. The technique avoids race conditions and guarantees that each device will recognise the synchronisation trigger on its next rising edge reference clock transition. The AD9852, after being reset, is configured to use an internally generated update pulse that is also placed onto its I/O-UD pin. The triggering can be configured to external with the I/O-UD pin as an input. When the system is powered up, the I/O-UD pin is therefore an output until the configuration is changed. The data sheet for the device does not describe the output architecture for this pin and so, as a precaution, a tri-state buffer is used to prevent the driving of this pin while it is configured as an output. Normally one would expect it to be initially configured as an input and/or have an open-collector output architecture. The buffer is activated by setting Q1 high and Q2 low of the latch U10. When the system is powered up the outputs of U10 are undefined and so two pins with opposite states are used to reduce the accidental enabling of the buffer.

The final part of this section is the RF analogue output associated with the AD9852 DDS. The AD9852 has differential current outputs whose maximum value is set by the resistor R47 and can be up to 20 mA. These two outputs are used to drive a 50 Ω balanced input transformer (T1) that has an unbalanced 50 Ω output. The transformer is a TC1-1T from Minicircuits [109]. A low-pass filter with a cut off frequency of 40 MHz is then applied using a passive 3rd order Butterworth filter structure consisting of C78, C83 and L3 (figure 5.25, top right). RF output pulses can be generated by either changing the output amplitude of the AD9852, or gating the RF output using the ADG901 RF switch, or both. The preference is to use both. Firstly the RF amplitude is set within the DDS and then the RF switch is used to gate the RF and therefore generate the pulses needed during the transmit phase of an experiment. During the receive phase the RF amplitude of the DDS is set to zero and the RF switch is turned off to minimise the amount of noise being fed into the RF power amplifier. Again two pins with opposite states are used to enable the RF switch thus preventing the accidental enabling of the RF switch during system power-up. The maximum RF output is set to 0 dBm (1.0 mW) by making R47 equal to 2900 Ω, equating to about 15 mA output drive from the AD9852.

The last section of the digital receiver schematics deals with the power supplies and the I/O interface to the DSP board and is shown in figure 5.26. The interface connections to the DSP are made via a 96-pin backplane connector with the same pin connections as the DSP board. Using the same pin connections allows for the easy construction of the back plane board. The board is powered by a single 7 V to 8 V DC feed from the backplane connector. This is then regulated down to 3.3 V and 5 V. The typical power supply demands are about 250 mA for the 5 V supply and 800 mA from the 3.3 V supply. For simplicity, linear regulators are used, but this means that the 3.3 V regulator dissipates around 3 W and so heatsinking is necessary. Power supply decoupling capacitors are also shown and altogether more than 100 capacitors are used.

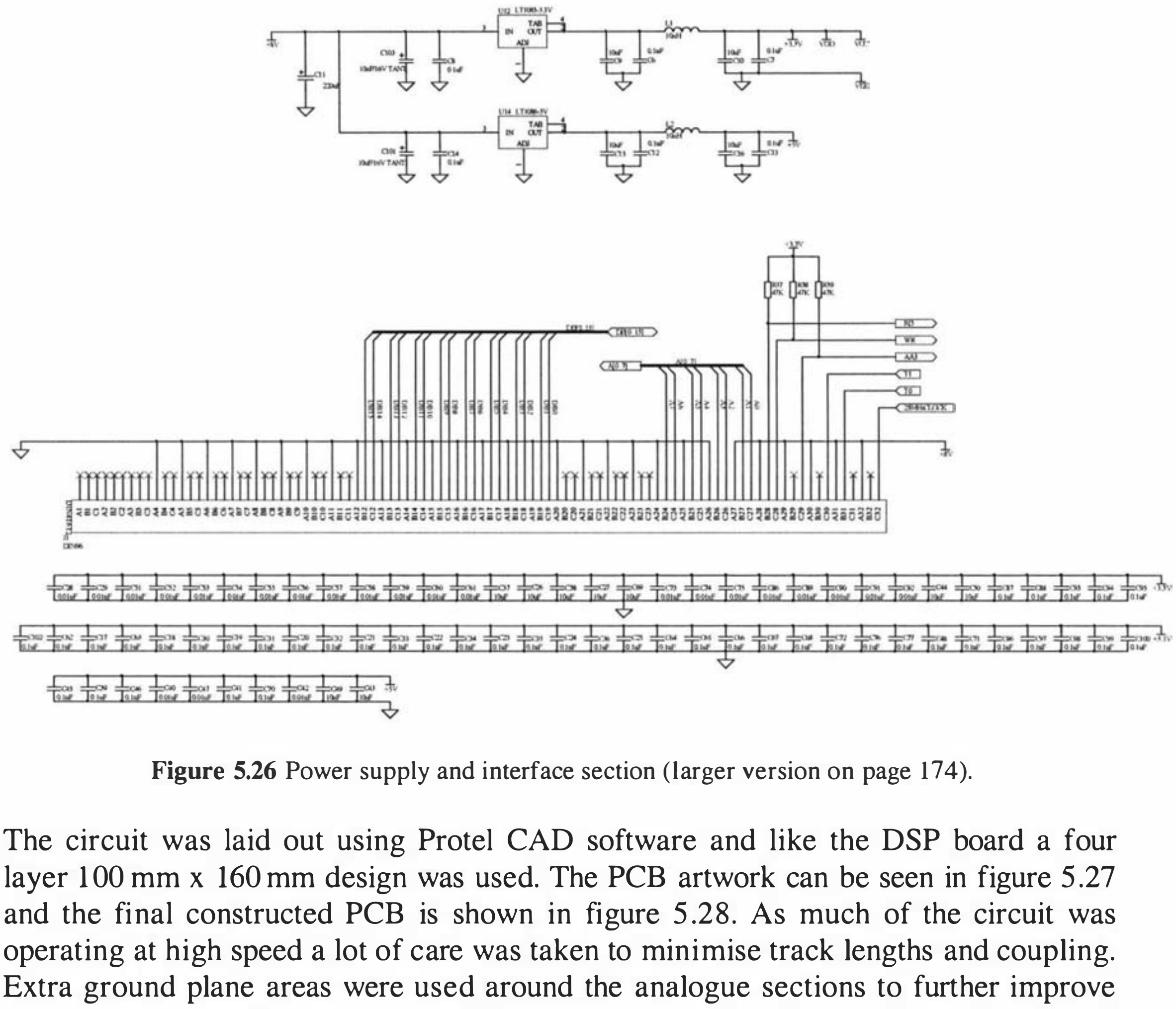


Figure 5.26 Power supply and interface section (larger version on page 174).

The circuit was laid out using Protel CAD software and like the DSP board a four layer 100 mm x 160 mm design was used. The PCB artwork can be seen in figure 5.27 and the final constructed PCB is shown in figure 5.28. As much of the circuit was operating at high speed a lot of care was taken to minimise track lengths and coupling. Extra ground plane areas were used around the analogue sections to further improve performance by reducing the ground impedance as well as the coupling of unwanted signals.

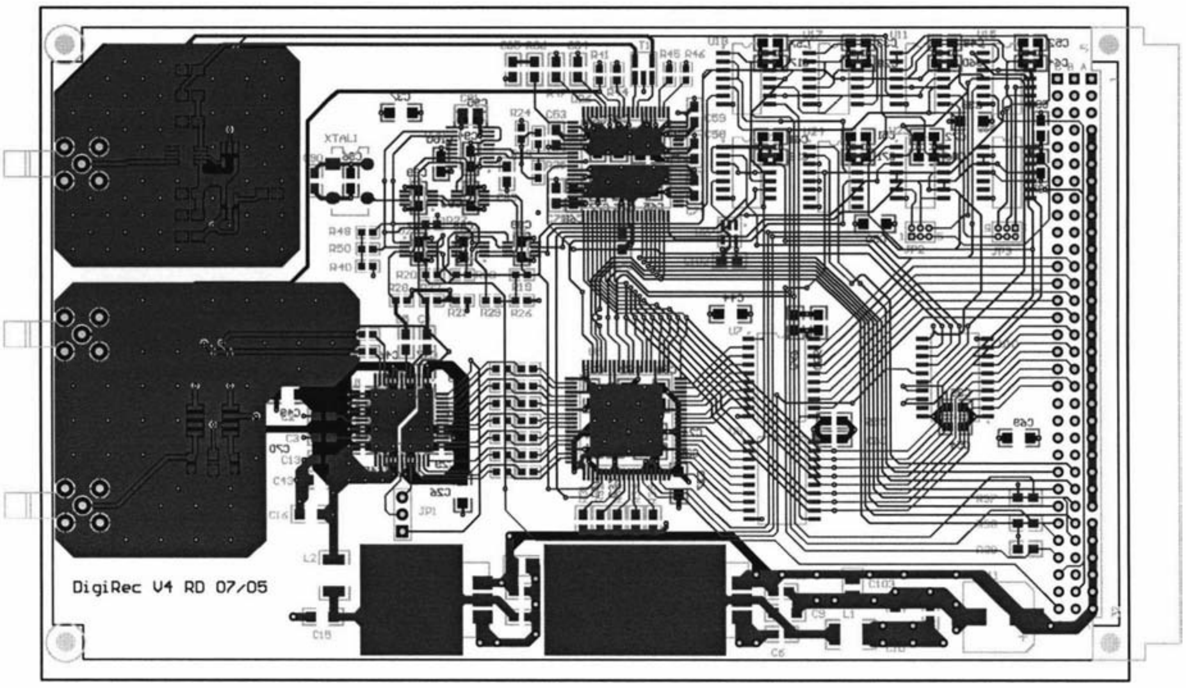


Figure 5.27 Digital transceiver PCB art work.

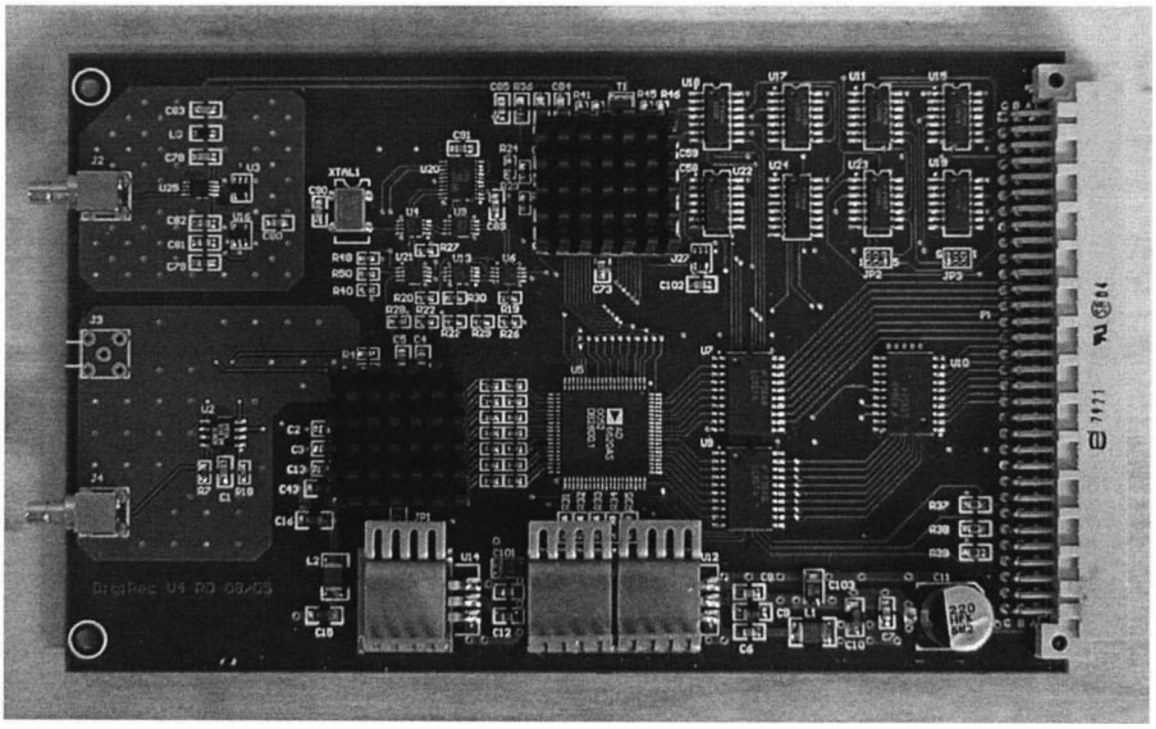


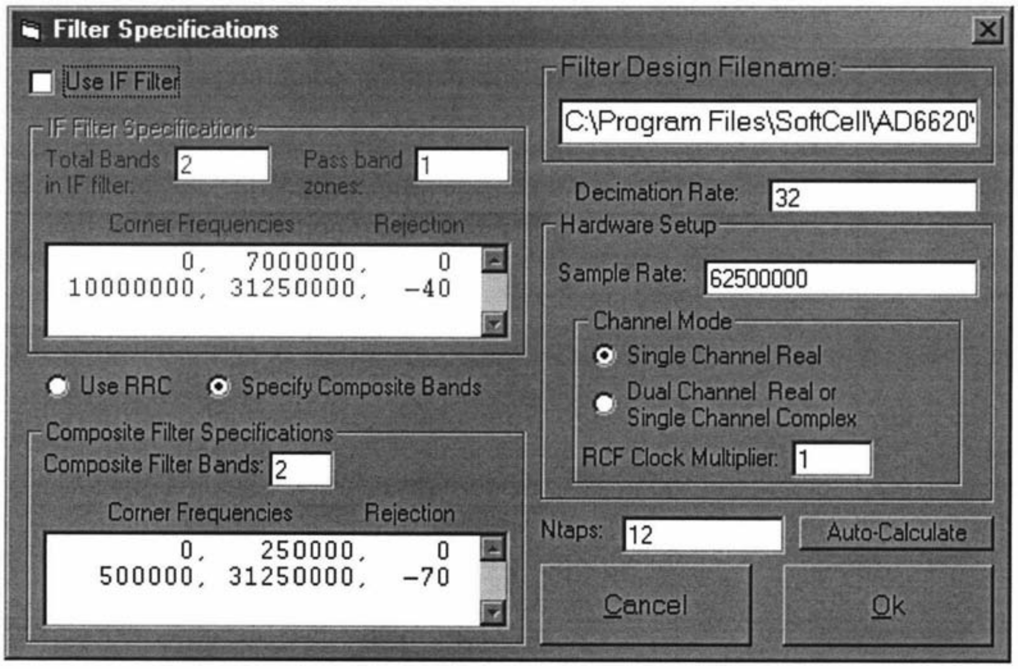
Figure 5.28 Constructed PCB.

5.3 Digital transceiver setup and software

The digital transceiver is not a stand-alone board. It must be connected to a DSP/USB board. A backplane board is used to provide the connections as well as the power supply feed. The NMR pulse programs always need to start with an initialisation sequence to configure and synchronise the transceiver. The programming sequence for this is as follows:

1. Synchronise DSP pulse program with 12.5 MHz clock.
2. Reset the transceiver devices.
3. Configure the AD6620 and load the digital filter coefficients.
4. Clear the existing data within the AD6620. This is to prevent previous data within the filter section from being combined with the new data.
5. Set up AD9852 parameters.
6. Start the AD6620 and AD9852 devices using a synchronisation pulse.

An example of this initialisation sequence can be found within the pulse program contained within appendix B1. The configurations of the AD6620 and AD9852 devices are determined by the receiver architecture, the type of probe that is used and the experiment. To begin with, the transceiver was configured for the NMR-MOUSE. The digital filter coefficients and the decimation rates for the AD6620 were calculated using the filter design software and the filter specification window is shown in figure 5.29. The receiver operates in a single channel real mode and therefore only processes one stream of data. The data stream is at the same rate as the AD6620's reference clock and so the clock multiplier is set to 1.


Filter Specifications

Use IF Filter: unchecked

IF Filter Specifications
Total Bands in IF filter: 2
Pass band zones: 1

```
Corner Frequencies        Rejection
        0,   7000000,       0
 10000000,  31250000,     -40
```

[Use RRC] [✓ Specify Composite Bands]

Composite Filter Specifications
Composite Filter Bands: 2

```
Corner Frequencies        Rejection
        0,    250000,       0
   500000,  31250000,     -70
```

Filter Design Filename: C:\Program Files\SoftCell\AD6620\

Decimation Rate: 32

Hardware Setup
Sample Rate: 62500000

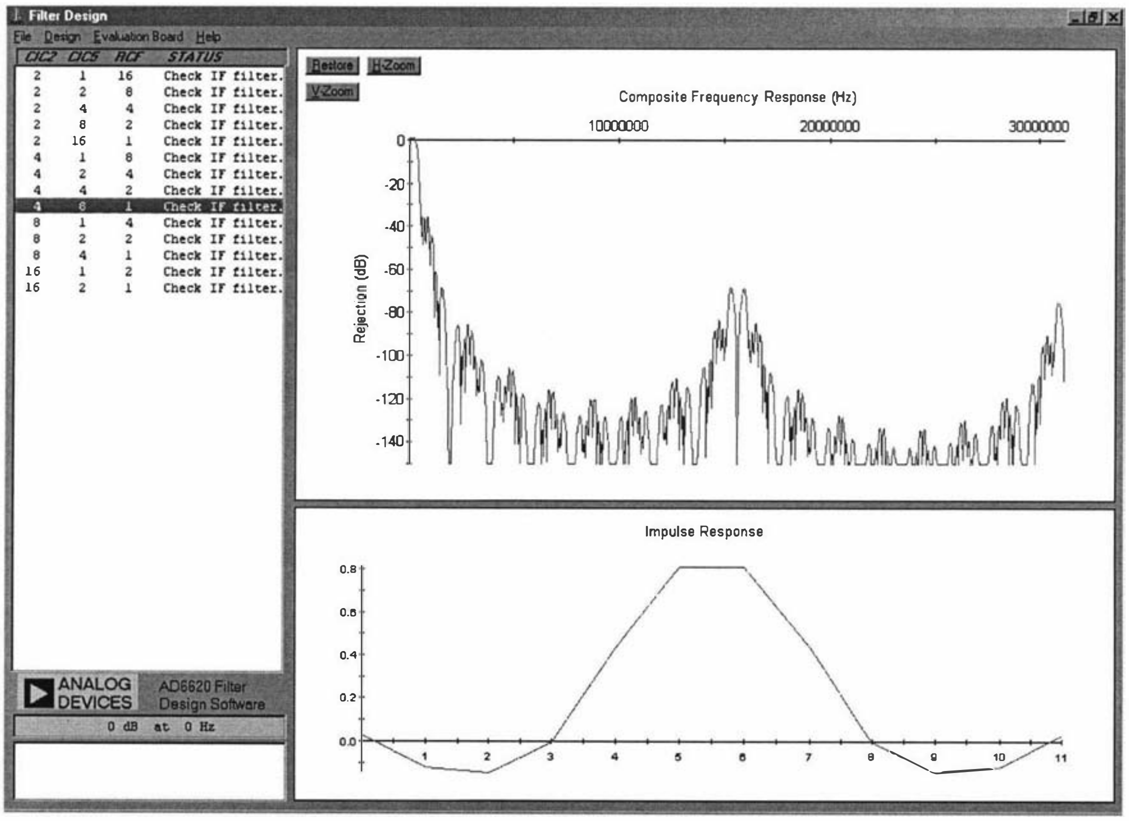
Channel Mode
[✓ Single Channel Real]
[Dual Channel Real or Single Channel Complex]
RCF Clock Multiplier: 1

Ntaps: 12

Figure 5.29 Filter specification window for NMR-MOUSE.

A total decimation rate of 32 was decided upon to give an output sample rate of just under 2 MHz. This would allow for the wide bandwidth that is necessary for the very short spin-echo signal. The number of taps for the final filter was limited to 12 for the same reason. As the signal is a pulse of short duration, the impulse response of the filter should also be of short duration in order to avoid dispersion. The filter can therefore be considered as a matched filter and consequently should be optimal in terms of signal to noise performance and distortion. However, with only 12 FIR filter coefficients the frequency response is rather poor and therefore decimation could not be used in the last section as the filter can not provide enough alias rejection. The pass-band/stop-band corner frequencies and attenuations were chosen to be 250 kHz with zero attenuation and 500 kHz with -70 dB.

The design software calculates the optimal responses for various combinations of decimation rates. It is wise to experiment with the various parameters and therefore perform many iterations before a final solution is chosen. Figures 5.30 and 5.31 show the frequency response of the chosen solution. The decimation rate combination used is 4 for the CIC2 section, 8 for the CIC5 section and 1 for the RCF section. The main issue when choosing the decimation combination is the rate of the CIC2 section. Here there is a trade off between device power consumption and alias rejection. The filter design software assumes that the signal has been shifted to base-band. The NCO frequency is set to the carrier frequency of the input signal and the two signals are then mixed digitally. The frequency response plot shown in figure 5.30 is referenced to the input frequency and so after decimation it would all get folded into a 0 to 977 kHz region (62.5 MHz/64). The minima are all periodically spaced at 1.95 MHz intervals, which is good as, after folding, they will coincide with the 0 Hz point on the response. This will result in fewer aliased frequency components interfering with the frequency band of interest.



CIC2	CIC5	RCF	STATUS
2	1	16	Check IF filter.
2	2	8	Check IF filter.
2	4	4	Check IF filter.
2	8	2	Check IF filter.
2	16	1	Check IF filter.
4	1	8	Check IF filter.
4	2	4	Check IF filter.
4	4	2	Check IF filter.
4	8	1	Check IF filter.
8	1	4	Check IF filter.
8	2	2	Check IF filter.
8	4	1	Check IF filter.
16	1	2	Check IF filter.
16	2	1	Check IF filter.

0 dB at 0 Hz

Figure 5.30 Receiver frequency response and FIR filter impulse response.

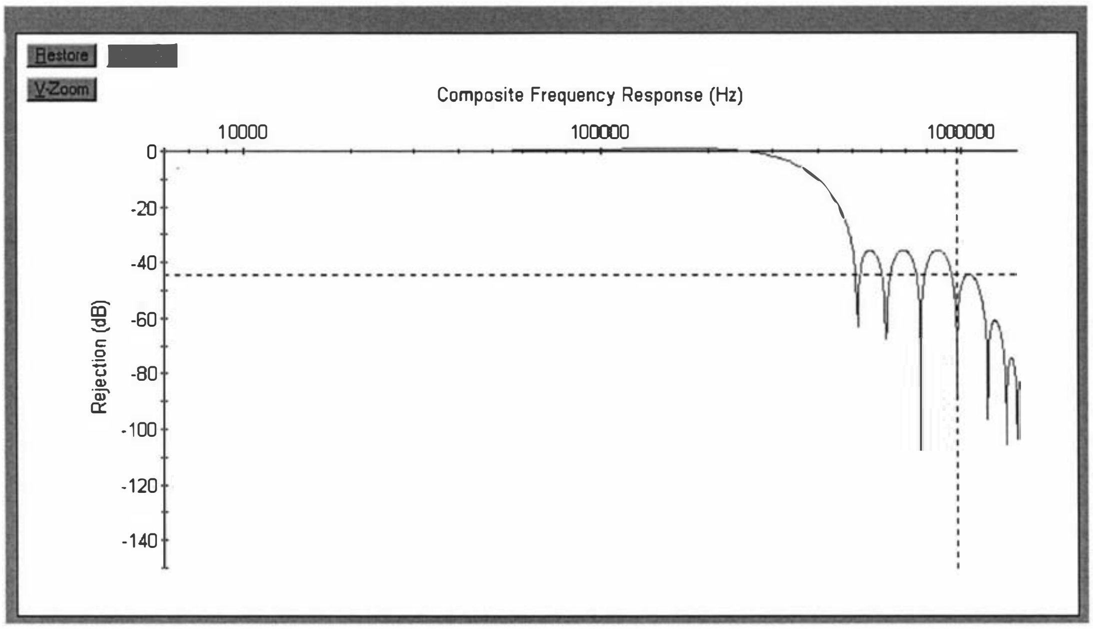


Figure 5.31 Zoomed receiver frequency response.

The response of the digital receiver can be improved by reducing the bandwidth of the analogue input signal. For this example it could result in the elimination of the spectral copies that occur at 15.625 and 31.25 MHz (figure 5.30) and increased rejection after 500 kHz (figure 5.31). This can be achieved by using a band-pass filter (IF filter). Allowance is made within the filter design software to specify any channel shaping, the software then uses the information to generate a composite frequency response. An example is shown in figure 5.32 where an analogue band-pass filter with a 1 MHz bandwidth is used with the chosen digital filters.

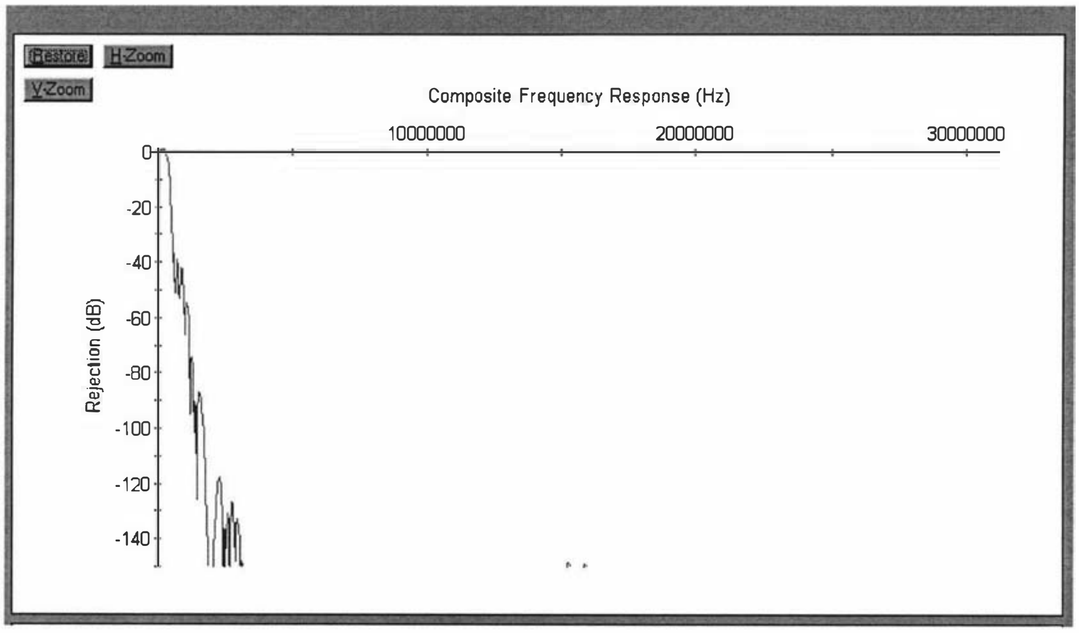


Figure 5.32 Frequency response of the receiver with analogue band-pass filter channel shaping. The amplitude of the spectral copies have been significantly reduced.

The filter design software can also produce a file that contains the filter coefficients in a DSP ready 20-bit hexadecimal format. These are listed, together with the actual values, as follows:

```
 0.0297      ( 3CDF)
-0.1222      (F05BA)
-0.1478      (ED13D)
-0.0055      (FF4B3)
 0.4351      (37B01)
 0.8107      (67C66)
```

The filter's impulse response is symmetric so only half of the 12 coefficients are shown. For a unity gain filter, these 6 coefficients should add up to 1.

The AD6620 has a series of inter-stage scalers (or attenuators) that need to be set in order to maximise dynamic range and yet prevent saturation. The scalers can be adjusted in 6dB increments. The first of these is on the output of the CIC2 section and the scaling coefficient (S_{CIC2}) is calculated using equation 5.2 and in this case equals 2. Where M_{CIC2} is the decimation rate of the CIC2 section and is set to 4, *Input_Level* is the fraction of the full scale input and is normally set to 1.

$$S_{CIC2} = \log_2\left(M_{CIC2}^{\,2} \times Input\_Level\right) - 2 \qquad (5.2)\ [93]$$

The maximum output level after the CIC2 scaler can be determined using equation 5.3 and in this case is equal to 1.

$$OL_{CIC2} = \frac{M_{CIC2}^{\,2}}{2^{S_{CIC2}+2}} \times Input\_Level \qquad (5.3)\ [93]$$

The output of the CIC5 section can also be scaled and its coefficient (S_{CIC5}) is calculated using equation 5.4 and is equal to 10, where M_{CIC5} is the decimation rate and was chosen to be 8.

$$S_{CIC5} = \log_2\left(M_{CIC5}^{\,5} \times OL_{CIC2}\right) - 5 \qquad (5.4)\ [93]$$

The output level after the CIC5 scaler is determined using equation 5.5 and again is equal to 1.

$$OL_{CIC5} = \frac{M_{CIC5}^{\,5}}{2^{S_{CIC5}+5}} \times OL_{CIC2} \qquad (5.5)\ [93]$$

The last filter section is the variable coefficient FIR filter and as no decimation is used M_{RCF} is set to 1. The output of this section can also be scaled by setting the S_{OUT} coefficient, but this time there is no equation to use as the output signal level is determined by the gain of the FIR digital filter. If the filter has unity gain then S_{OUT} should be set to 4, which equates to unity gain. The scaling values should be checked by applying an input signal to the receiver and checking to see that the data is not saturated or too small.

The AD9852 DDS is configured to minimise power consumption by operating in the single frequency mode with the inverse SINC filter turned off. External update clocking (synchronisation) is also selected. The other parameters such as the NCO frequency, NCO phase and the output amplitude are determined by the system user and can be altered during a pulse program. This allows for the generation of RF pulses with various phases and amplitudes.

5.4 Transceiver testing

The transceiver was then connected to the DSP/USB board so that tests could be performed to see if the system was functioning as expected. These tests would verify both hardware and software reliability and capabilities. A backplane circuit board was used to provide the interconnections between the transceiver and the DSP/USB system core. Software was written for both the DSP and Prospa to perform the tests. The backplane as well as the software will be explained later when a complete NMR spectrometer system is described.

5.4.1 Transmit section testing

The transmit section, which is the section that is based on the AD9852 DDS, was the first to be tested as it would be needed to test the digital receiver. A pulse program was written to first configure the AD9852 and then to generate a 1 MHz frequency burst. The pulse program also produced a trigger pulse on a TTL output so that an oscilloscope could be triggered to capture the frequency burst. A 50 Ω load was connected to the output of transmit section to provide the correct load impedance. The result of the first test is shown in figure 5.33.

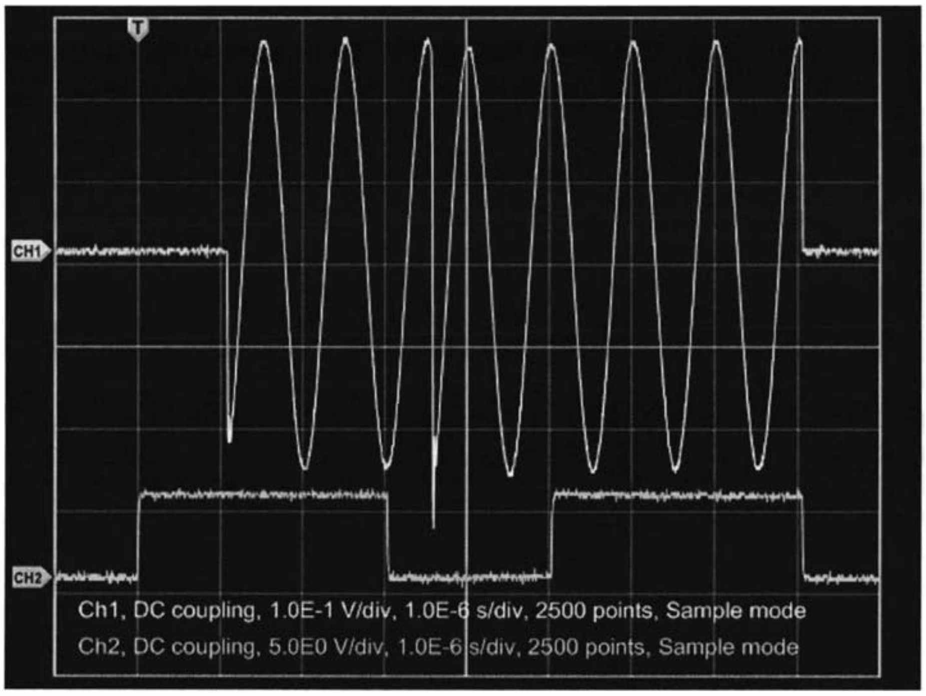


Figure 5.33 A 1 MHz transmit burst with a 180 degree phase transition. The lower trace shows the TTL trigger pulse.

Here the DDS was programmed to produce a 1 MHz burst that included a 180 degree phase transition. The phase transition can be seen at approximately 500 ns after the end of the first TTL pulse. The command to the DDS to change its phase is sent just after the end of the first TTL pulse. The ADG901 analogue switch was turned on for the whole of the test so that DDS output could be completely observed. The bottom trace shows the TTL line from the DSP/USB board that was used for triggering. This line would normally be used to control the blanking of an RF amplifier. The line level goes high 1 μs before the start of the burst and for a working spectrometer this is done to allow time for the RF power amplifier to settle. In figure 5.33, two TTL pulses can be seen and this is due to the pulse program being written primarily for a spin-echo experiment.

As well as phase, the DDS output amplitude can also be changed and this is shown in figure 5.34. The final parameter that can be changed is the frequency. This is demonstrated in figure 5.35, where now the frequency, amplitude and phase are all changed within 1 μs from the end of the first TTL pulse. The oscilloscope was also set to averaging and 20 experiments were performed to demonstrate the repeatability and hence the timing and synchronisation stability of the transmitter section.

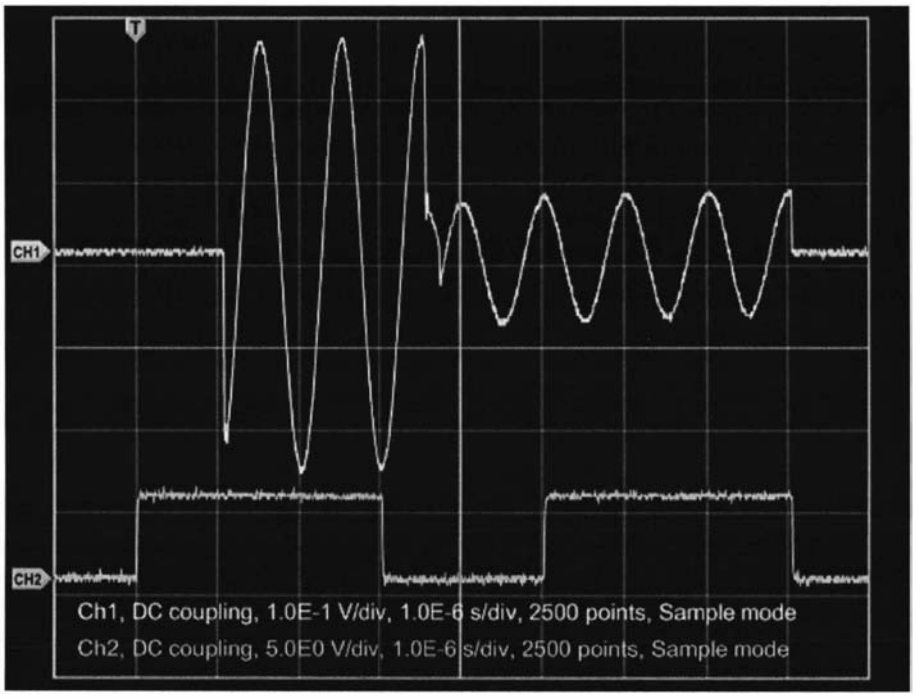


Figure 5.34 A 1 MHz transmit burst with amplitude and phase change.

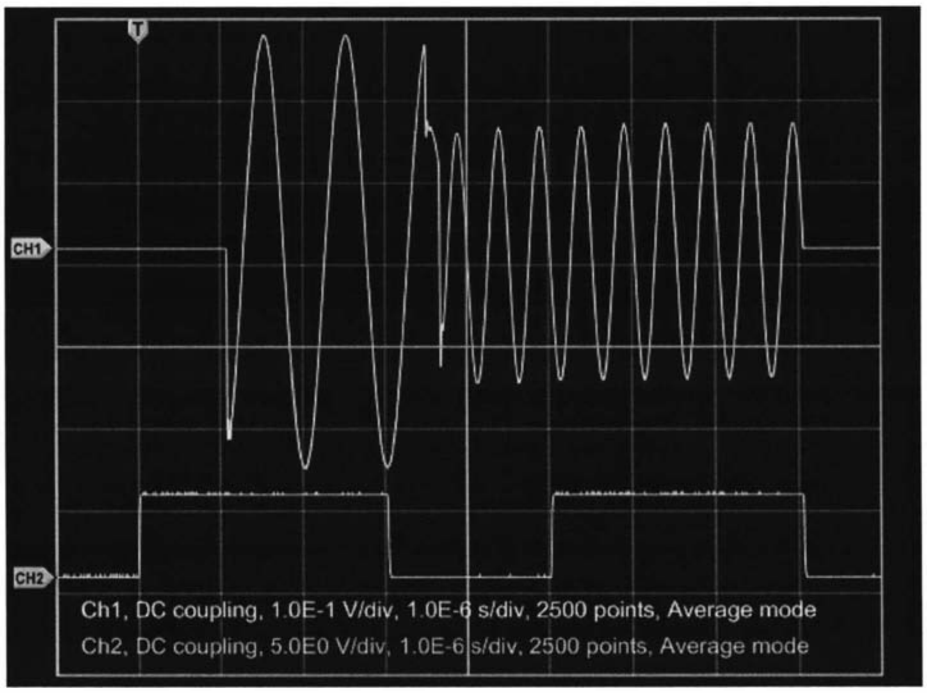


Figure 5.35 Transmit burst with amplitude, frequency and phase changed. 20 scans were averaged to demonstrate the repeatability of the transmitter section.

It is interesting to note that when the AD9852 DDS parameters are changed the output amplitude changes first. This is due to the amplitude control unit within the DDS being towards the end of its processing pipeline. The AD9852 data sheet [98] specifies that after an update it will take 9 clock cycles for a change in amplitude to occur. In this case, using a 125 MHz clock, this would equate to a delay of 72 ns. The pipeline delay for the DDS core is 33 clock cycles (264 ns). This would therefore be the delay after an update pulse in updating the frequency or the phase.

The pulse program was then modified to include the ADG901 RF switch and to set the DDS output to its maximum amplitude and a frequency of 2 MHz. Figure 5.36 shows the result where now the continuous output of the DDS is gated by the RF switch to produce two bursts. The result shows that the ADG901 performs well with good *off* isolation and no transients during switching. The output voltage level of the bursts was measured to be 624 mVpk-pk, which is just under 0 dBm (632 mV). The output level can be trimmed to the desired level of 0 dBm by adjusting R47. The slightly reduced amplitude would be due to the insertion losses of the RF switch and the low-pass filter within the transmit section.

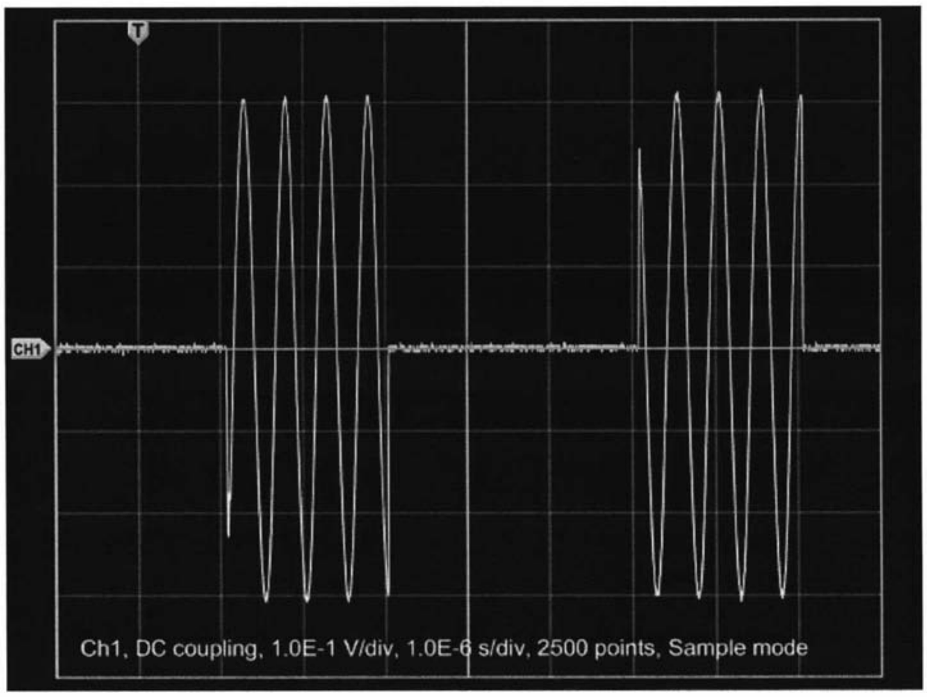


Figure 5.36 A 2 MHz transmit burst at full amplitude and with RF switching.

The test was then repeated but this time the DDS was set to 20 MHz. The result is shown in figure 5.37. Note that the variation in the amplitude is due to the finite resolution of the oscilloscope. This test was then repeated 20 times with the oscilloscope set to average 16 scans and to also only show the initial 400 ns of the first burst. Like the previous tests, the TTL pulse was used as the trigger source. The result is shown in figure 5.38. This result demonstrates the stability of the synchronisation of the DDS and the DSP pulse program.

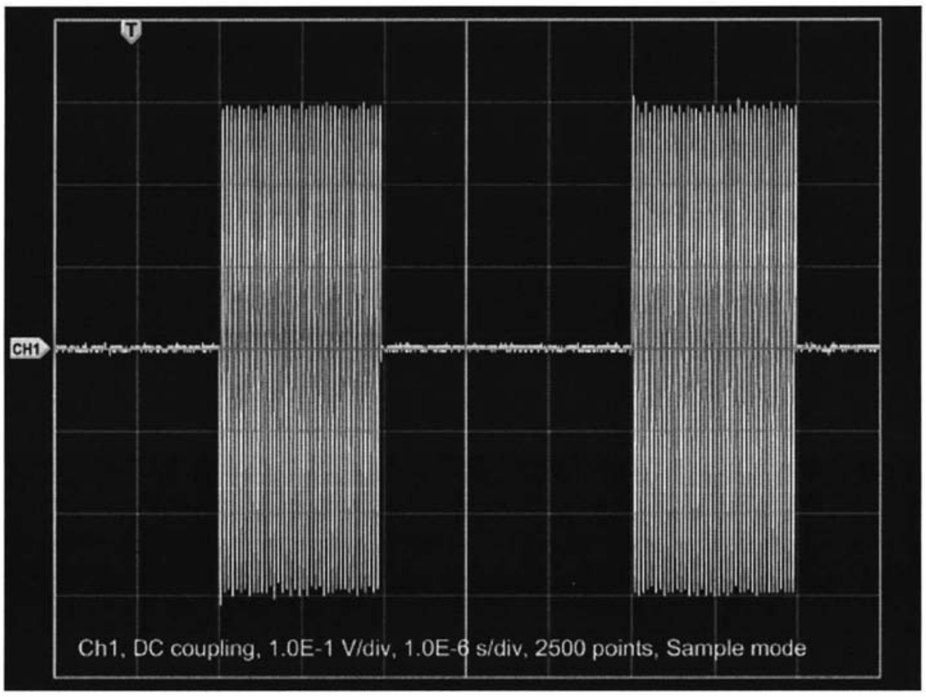


Figure 5.37 Transmit burst using 20 MHz, full amplitude and RF switching.

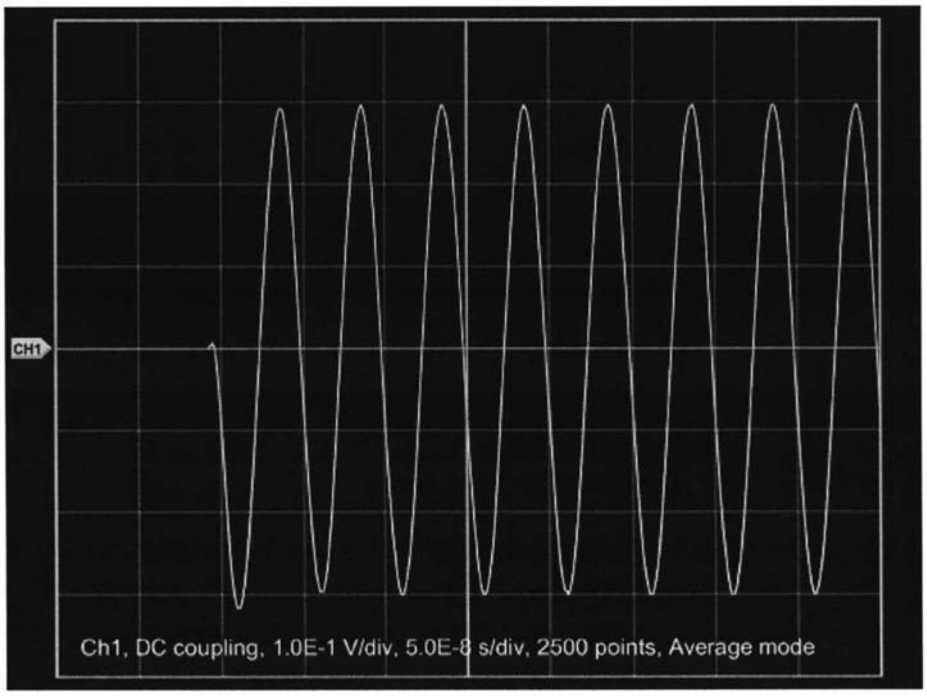


Figure 5.38 Zoomed first part of the transmit bursts. Again 20 oscilloscope scans were averaged to demonstrate the repeatability of the transmitter section.

5.4.2 Receive section testing

Now that the DDS was found to be working correctly it could then be used as a frequency source to test the digital receiver. This was necessary to test the synchronisation of both parts of the transceiver. For the following tests a pulse program was written that first set up the DDS to provide a continuous reference signal and then set up the digital receiver to capture some data. The RF switch was also left in the on state. The first test of the receiver section was simply applying the near 0 dBm sine-wave signal from DDS to the input of the receiver and then observing the output data. The DDS was set to 20.1 MHz and the NCO within the AD6620 was set to 20 MHz and so, after digital mixing, the resultant data should be a sine-wave at 100 kHz. This is what was observed and is shown in Figure 5.39.

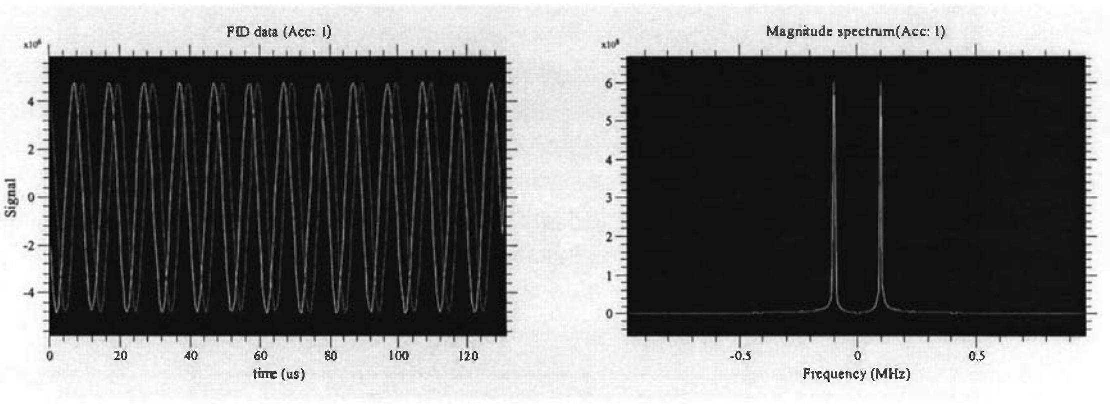


Figure 5.39 Testing of the digital receiver. A 20.1 MHz signal from the DDS was applied to receiver with the NCO set to 20 MHz. The rest of the receiver parameters were configured as per chapter 5.3. The left hand plot shows the *I*nphase and *Q*uadrature time domain data obtained from the digital receiver. The right hand plot shows the magnitude spectrum that was obtained by performing a Fourier Transform of just the *I*nphase (or real) data.

For a DDS amplitude of 624 mVpk-pk, a receiver input range of 1.1 Vpk-pk and 24-bit output data resolution we would expect the peak to peak amplitude of the data to be:

$$(624\,\text{mV}/1.1\text{V}) \times 2^{24} = 9.52 \times 10^{6} \qquad (5.6)$$

The amplitude of the time domain data in figure 5.39 was read as 9.54×10^{6} ($\pm 0.08 \times 10^{6}$) so the calibration of the receiver can be considered to be correct. This would indicate that the receiver scaling and filter coefficients had been correctly determined. Note that the peak heights in the frequency domain plot are 128 times the amplitude of the 256 data point sine-wave signal in the time domain.

The test was then repeated 10 times with signal accumulation (averaging) enabled to check the synchronisation of the DSP, DDS and receiver. The result is shown in figure 5.40. It can be seen that both the time and frequency domain amplitudes are a factor of 10 greater than those shown in figure 5.39 indicating that the received data coincided and therefore proving that the sections were synchronised.

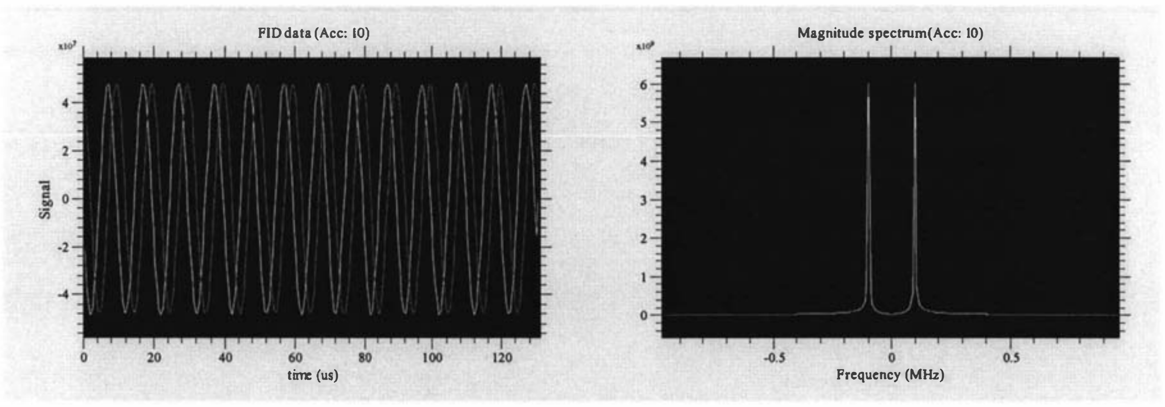


Figure 5.40 Further testing of the digital receiver. This time the test was repeated 10 times with signal accumulation enabled. As expected, the time and frequency domain amplitudes/magnitudes are10 times greater than those in figure 5.39.

The frequency response of the receiver was then determined to check if the filtering was functioning as designed in chapter 5.3. This was done by varying the offset frequency between the DDS and the receiver and then measuring the amplitude of the output data from the receiver. Figure 5.41 is the response and follows the theoretical response that was calculated earlier and was shown in figure 5.31. The response was only determined up to 500 kHz as beyond that it was difficult to measure. This was due to the data becoming very small and non-sinusoidal.

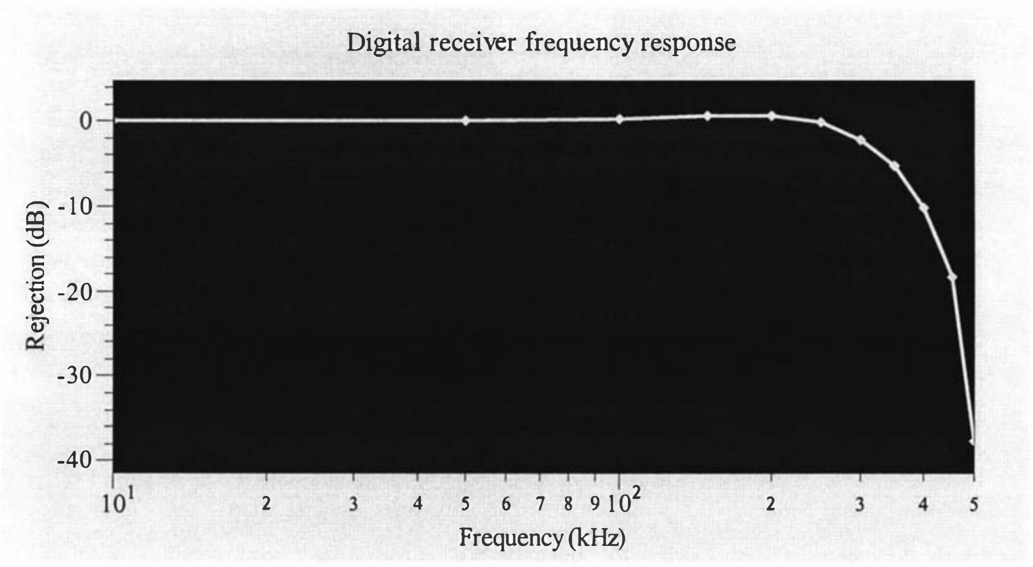


Figure 5.41 The measured frequency response of the digital receiver.

The final test of the receiver was to determine the signal to noise performance. A 50 Ω terminator was connected to the input of the receiver in place of the signal generator. The receiver NCO was set to 20 MHz and another set of data was obtained. This is shown in figure 5.42. Here we can see the baseline noise of the receiver as well as the response of the digital filtering.

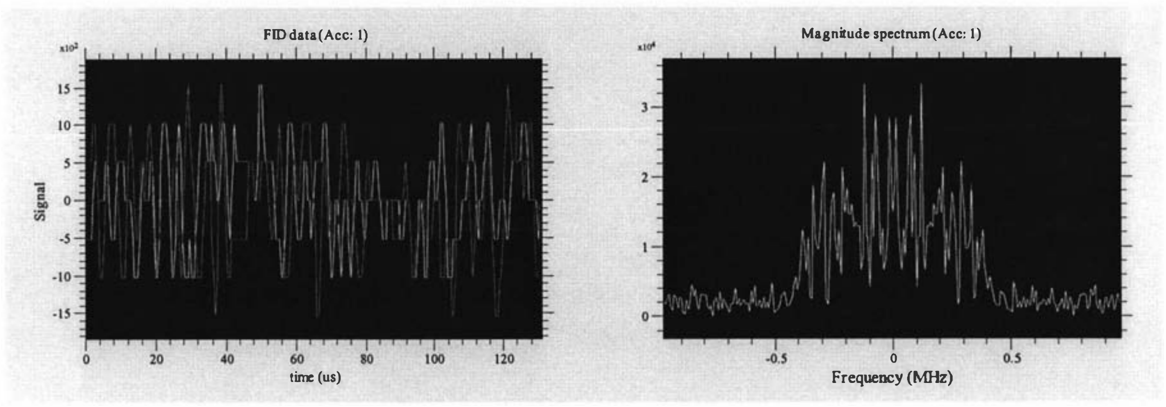


Figure 5.42 Baseline noise of the digital receiver.

The input bandwidth of the receiver is very broad and is approximately 250 MHz and so for a 50 Ω terminating resistor the thermal noise level would be 14.4 μV rms. The receiver input resolution, which is determined by the 14-bit AD6644 ADC is 67 μV which is significantly greater than the noise contributed by source resistance and so we can safely ignore it. The rms value of one of the time domain quadrature channels was calculated from the data to be 602 (corresponding to 39 μV rms). The 16-bit data from the digital receiver is placed in the upper bits of a 24-bit word. Therefore the maximum amplitude that a signal could have would be $\pm 2^{23}$, which equates to an rms value of 5.93×10^{6}. The maximum signal to noise ratio can therefore be determined as 9852 or 80 dB. Another method for determining the signal to noise ratio of a spectrometer, that is popular among chemical spectroscopists, is to use the ratio between the spectral height of the signal compared to the height of the spectral noise floor. So using the data from figures 5.39 (6×10^{8}) and 5.42 (3.4×10^{4}) a S/N ratio of 17647 or 85 db can be obtained. Note that the S/N ratio could still be higher if the signal within figure 5.39 reached the maximum possible values. This method of using the heights of the spectral peaks is very quick, but is not one that is preferred.

A S/N ratio of 80 dB is certainly adequate for portable NMR systems as the total system's S/N performance will be severely limited by the S/N performance of the probe which is often under 10. The time domain noise data showed only 6 discrete levels which would equate to less than three bits. The output from the receiver is 16-bit data and so overall the performance is greater than 13 bits, which is good considering that the resolution of the input ADC was 14 bits. The receiver filter had a cut off frequency of approximately 500 kHz and so the receiver could be considered to be broadband. The S/N performance could be improved by narrowing the filter bandwidth, but this could result in a short spin-echo signal becoming distorted (or dispersed). Also, no filtering was applied to the input of the receiver and so it was operating with a full analogue bandwidth of approximately 250 MHz. The sources of noise are the input amplifier, the aperture jitter of the sampling clock and interference being picked up in the analogue input circuitry. The noise due to the aperture jitter will increase as the signal increases in amplitude or frequency and so the final S/N performance will be less than 80 dB, however figure 5.39 showed very little noise in the frequency plot when a large amplitude signal was applied.

6.0 RF front end

6.1 Introduction

The digital transceiver can provide and process all the signals within an NMR system, but only for a narrowly defined range of signal amplitudes. Therefore, before the transceiver can be connected to the probe some other modules are required to provide the appropriate signal levels. These modules are called the "RF front end" as they deal directly with the probe and usually consist of a transmit RF power amplifier, a very sensitive and low noise preamplifier, a variable gain receiver amplifier and a RF transmit/receive switch (duplexer). A typical arrangement of the modules is shown in figure 6.0.

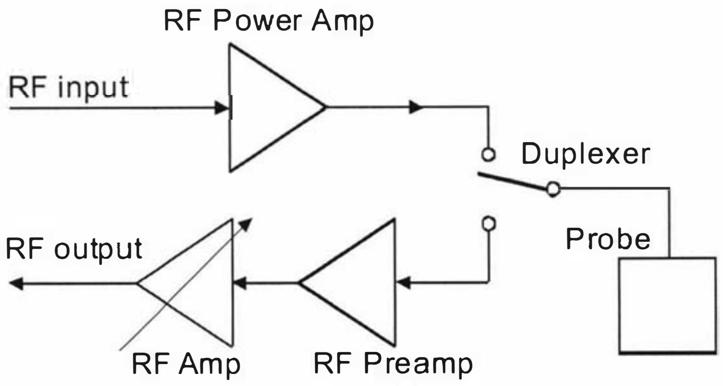


Figure 6.0 RF front end block diagram.

Probes operate at different frequencies and with various RF transmit power levels and so it is difficult to produce modules for a wide range of applications. Therefore the RF front end is considered as application specific, unlike the DSP/USB board and digital transceiver. However, the variable gain receiver amplifier module within the RF front end can be used for most applications if it is designed to operate as a broadband amplifier. This approach was chosen to minimise the effort when modifying the system for new applications and hence a digitally-controlled variable-gain broadband receiver module was designed. The developers of the NMR-MOUSE have also produced RF power amplifier and preamp/duplexer modules to support their probe and a set of these modules was obtained and interfaced to the system. This was a faster and easier way to get a complete NMR system and allowed the early evaluation of the system's performance. For the NMR-MOLE probe a simple RF front end was put together using a commercial RF power amp and preamplifier. Again this sped up the construction and evaluation of the system. It is intended that in the future a set of RF modules will be designed to support the various probes that are produced.

6.2 Variable gain receiver amplifier

The design of the variable gain receiver is based on the Analog Devices AD8369 [110] amplifier. This amplifier has a frequency range from about 1 kHz to 600 MHz and a variable gain range from -10 dB to +35 dB. A schematic of the device is shown in figure 6.1. The setting of the gain is achieved through the use of either a 4-bit parallel input or a three-wire serial interface and the device operates with both differential inputs and outputs.

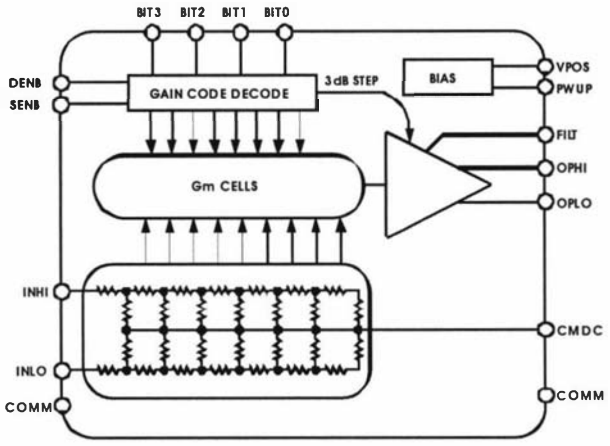


Figure 6.1 AD8369 variable gain amplifier functional diagram. Figure taken from reference 110.

The receiver amplifier module was designed to be a standard 100 mm x 160 mm sized PCB board and the schematic is shown in figure 6.2

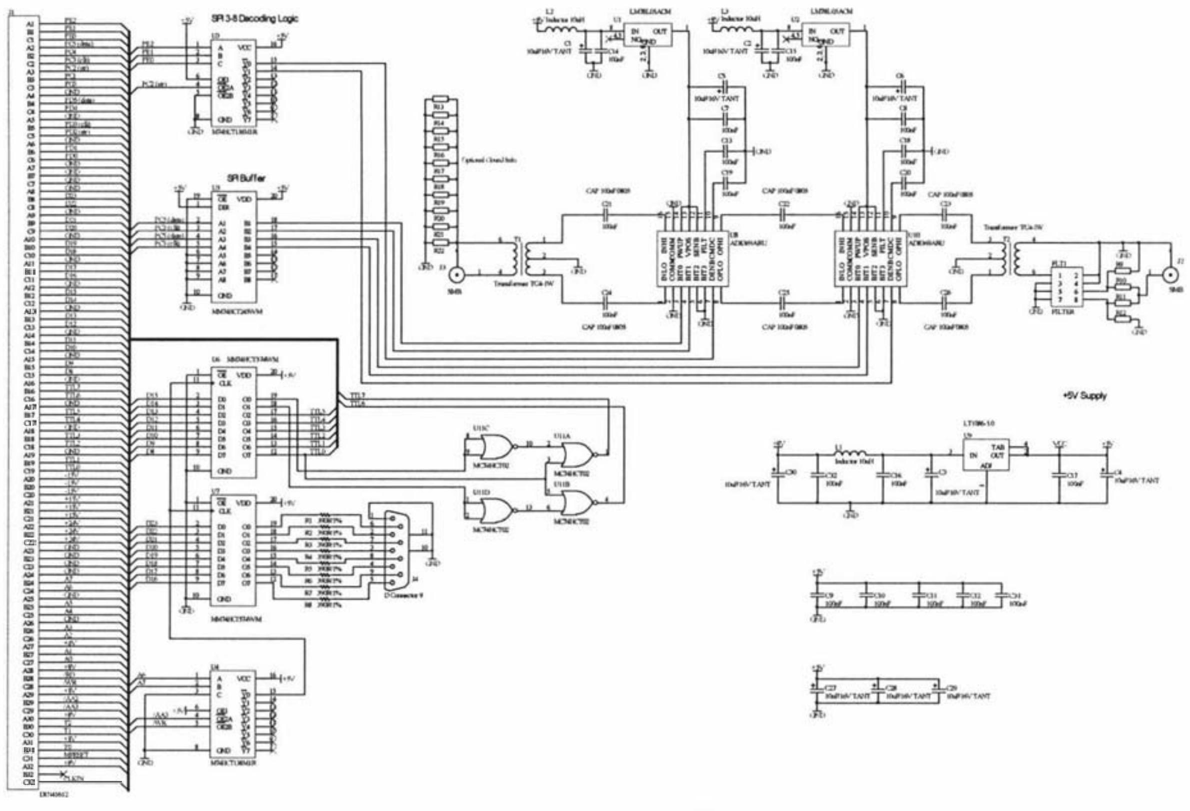


Figure 6.2 Variable gain receiver amplifier module.

The modules in the RF front end need some TTL control lines so two 8-bit data latches were included. The latches connect to the DSP/USB board data lines D8-D23 via a backplane connector and act as write only memory locations. The address of the latches is the same as for the 8-bit latch used within the digital receiver module. That latch was connected to D0-D7 so when the DSP writes to address $20000, the lower 8 bits go to the digital transceiver board and the upper 16 bits go to the receiver amplifier board. The outputs of the latch connected to D16-D23 are brought out onto a 9 pin D connector and are made available as general purpose TTL outputs. 390 Ω resistors are put in series with the outputs to prevent overloading. The outputs of the latch connected to D8-D15 are routed back onto the backplane connector and are intended to be used by the other RF front end modules. During the acquisition of an NMR signal it is necessary to minimise the residual output of the RF amplifier. Therefore RF power amplifiers require TTL signals to control the operation of the amplifier's output stage and two TTL lines (TTL6 and TTL7) are allocated for this task. TTL0 and some extra logic is included to safeguard against the accidental enabling of the RF power amplifier during system start up.

The receiver amplifier should have a maximum gain of at least 60 dB so two AD8369 devices are cascaded to provide a gain of 70 dB. The gain is set using one of the DSP/USB board's synchronous serial interfaces. These lines are buffered with separate outputs for each amplifier device to minimise crosstalk. The serial bus can be used for other devices within the NMR system and U3 is used to decode the lines PE0-PE2. The board is powered from the 7 V backplane supply which is regulated down to 5 V for the logic devices. Each of the amplifier devices has its own 5 V regulator to minimise parasitic coupling between them and therefore avoid oscillations. The arrangement of the amplifier devices can be seen more clearly in figure 6.3. The AD8369 devices are operated with 200 Ω differential inputs and outputs and so transformers (T1 and T2) are used to convert to 50 Ω. The two amplifier stages are coupled to each other and the transformers using 0.1 μF AC coupling capacitors. Since transformers are used, the amplifier input and output connections can be differential. This is an advantage when using multiple modules as often Earth loops can occur. If necessary one side can be tied back to ground using links R13 to R22. These links are distributed along the PCB signal line and when used ensure that the line is well connected to the ground plane.

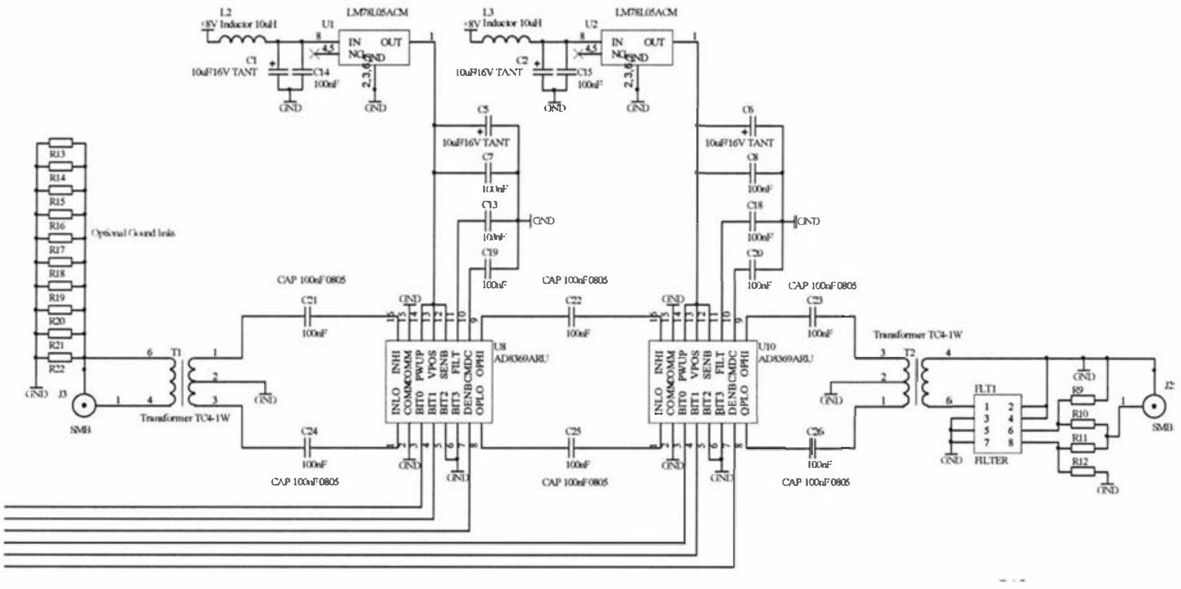


Figure 6.3 Arrangement of cascaded amplifiers.

The digital receiver requires that the input signal be band limited. Therefore provision is made to apply filtering at the receiver amplifier output. The filter would need to be chosen for each application so the circuit provides support for a wide range of Minicircuits filters. The filters have a common pin configuration, except for the output pins. Either pin 6 or 8 is used as the output, and if it is not an output it must be tied to ground. Four optional links (R9, R10, R11, R12) are used to provide the two options. A two layer design with extensive ground planes was used. The composite artwork is shown in figure 6.4 and the assembled board is shown in figure 6.5. The amplifier devices are each housed in individual screening cans. This minimises the coupling of interference and further reduces the coupling between the devices.

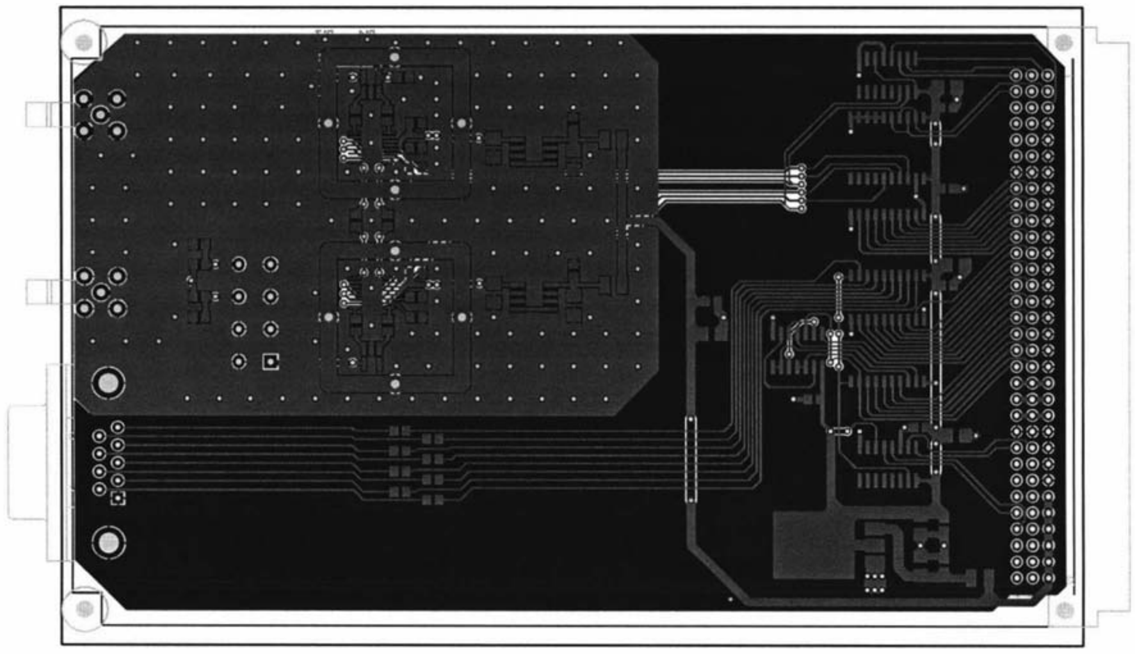


Figure 6.4 Receiver amplifier PCB artwork.

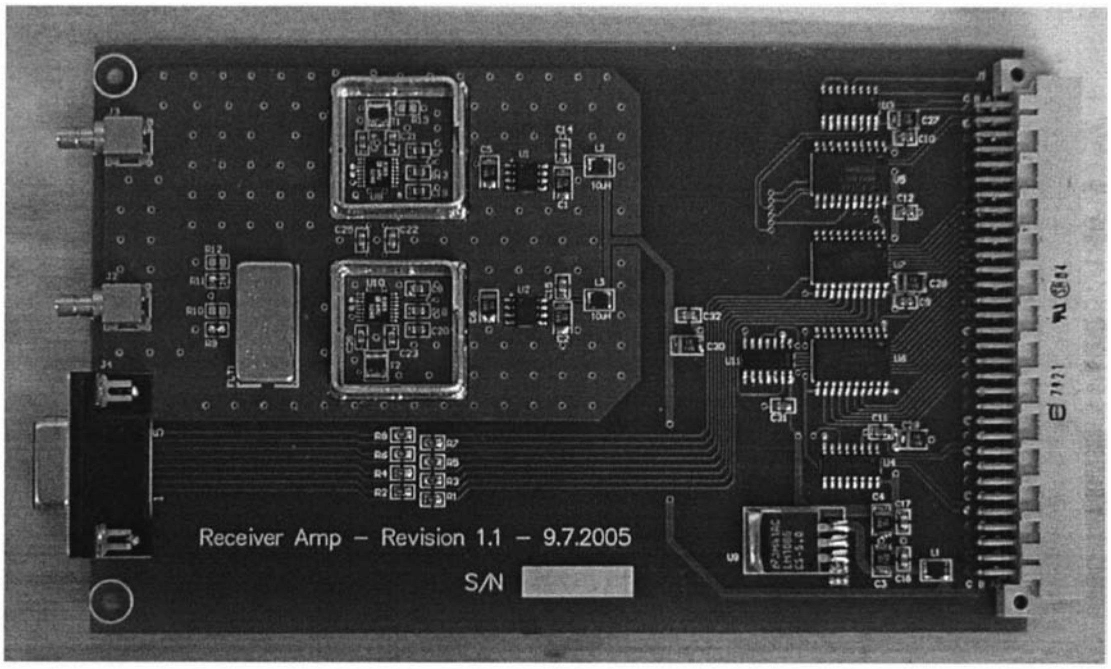


Figure 6.5 Receiver amplifier assembled board

6.3 NMR-MOUSE RF front end

The manufacturer of the NMR-MOUSE also provides an optimised RF front end to support their probe. It consists of a 250 W RF power amplifier module and another module containing the preamplifier, duplexer and tune/match circuit. A block diagram is shown in figure 6.6 and a picture of the modules together with a probe is shown in figure 6.7

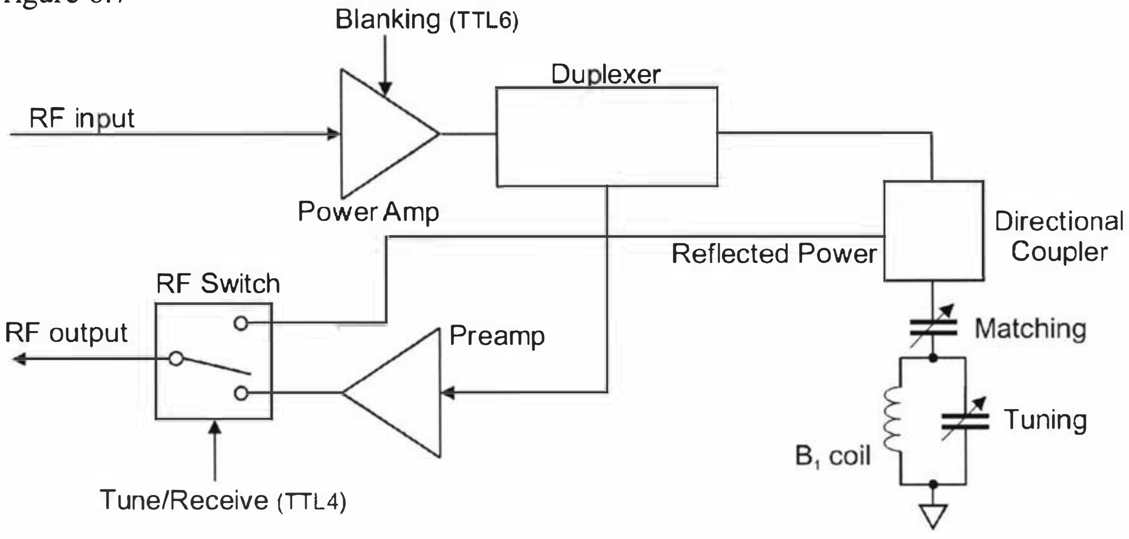


Figure 6.6 NMR-MOUSE RF front end block diagram.

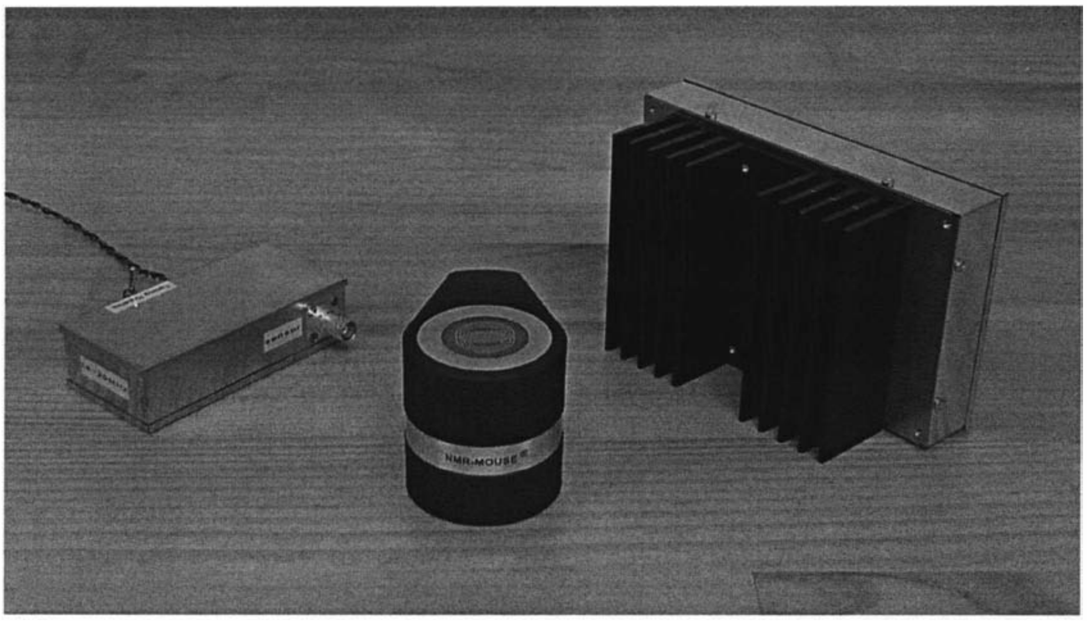


Figure 6.7 NMR-MOUSE with RF front end [22].

The RF output of the digital transceiver is connected to the input of the RF power amplifier and the output of the RF tune/receive switch is connected the input of the variable gain RF amplifier. The RF power amplifier has a blanking capability where the final high power output stage of the amplifier can be shut down. This is necessary to minimise the output noise of the amplifier during the receive phase of an experiment. A TTL input controls this and a low level blanks the amplifier. The duplexer is the transmit/receive switch which automatically switches into transmit mode when it senses any RF power. Being automatic always provides protection for the very delicate preamplifier. This RF front end also has a built in tuning/matching circuit so that the probe can be "wobbled" while remaining attached to the system. During wobbling, the system provides a low level frequency swept signal to the probe and a directional coupler is used to sense any reflected power. It is important that the probe be tuned to the resonant signal and matched to 50 Ω as any mismatch will result in poor signal to noise performance of the probe and could also lead to the damaging of the preamplifier. If the high RF power can not be absorbed by the probe due to mismatch then it will have to be absorbed elsewhere. A TTL line is used to select between the tune/match and receive mode.

6.4 NMR-MOLE RF front end

The NMR-MOLE probe and its applications are still in development and the final RF front end requirements are yet to be determined. Therefore, at this stage, it was pointless to consider designing custom modules so some standard off the shelf modules were purchased. A 250 W, 250 kHz-8 MHz RF power amplifier was purchased from Tomco [111] and a picture of it is shown in figure 6.8.

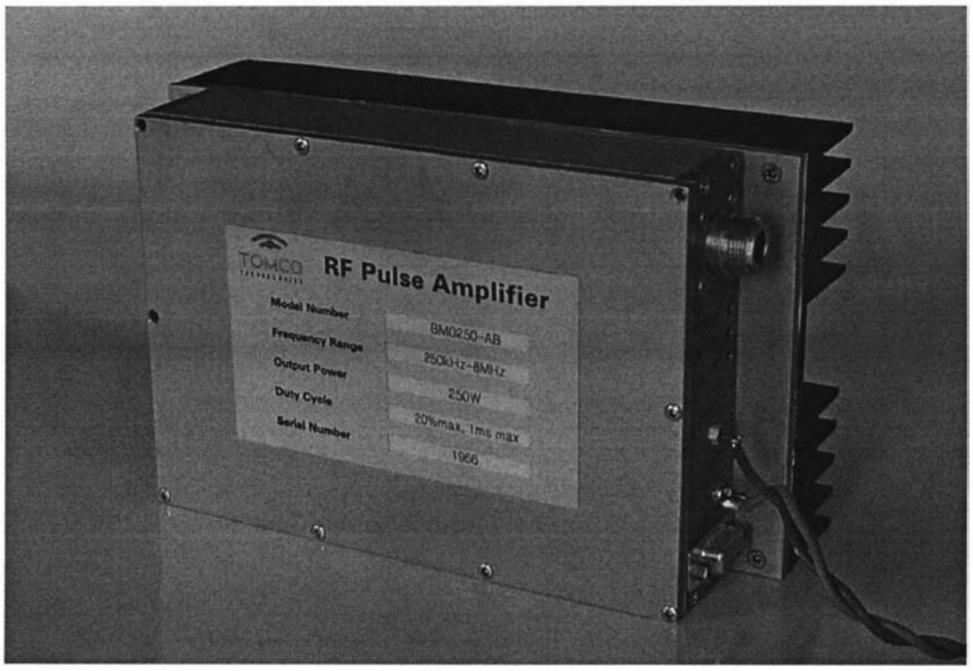


Figure 6.8 Tomco 250W RF power amplifier [111].

The amplifier operates in class AB and is designed for pulse applications with the maximum allowable pulse width of 1 ms. The amplifier requires a 48 V power supply, a 0 dBm signal for maximum RF output and a TTL signal for blanking.

An AU-1583 low noise preamplifier was purchased from Miteq [112] and is shown in figure 6.9. It has a typical noise figure of 1.2 dB, a gain of 36 dB and a bandwidth from 20 kHz to 400 MHz.

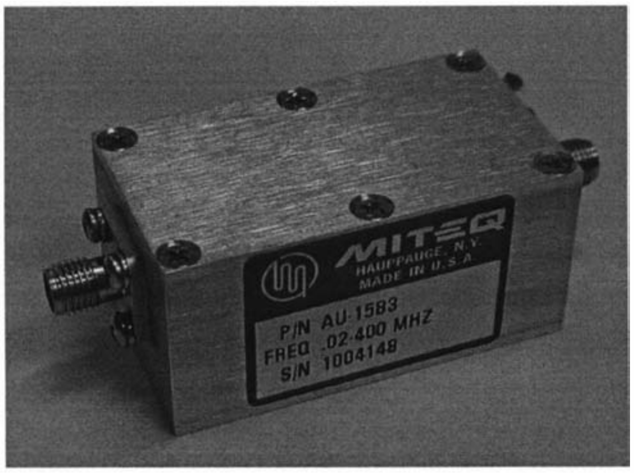


Figure 6.9 Miteq preamplifier [112].

A duplexer was designed and constructed using the common technique of a quarter wave line with crossed diodes, this is shown in figure 6.10. The diodes D1 are included to reduce the coupling of noise from the power amplifier during the receive phase of the experiment. The diode that was used for this purpose is the BAV21 from Philips [113]. During the transmit phase, the diodes D2 will conduct due to the high voltages present at the output of the RF amplifier. They will therefore act as a short on the end of the quarter wave line which will result in the other end of the line becoming a high impedance input. This high impedance then prevents any power from reaching the input of the preamplifier thus protecting it. During the receive phase, the quarter wave line acts as a low pass filter. 1N4148 diodes were used for D2, they are fast, have low capacitance and can be obtained from many manufacturers.

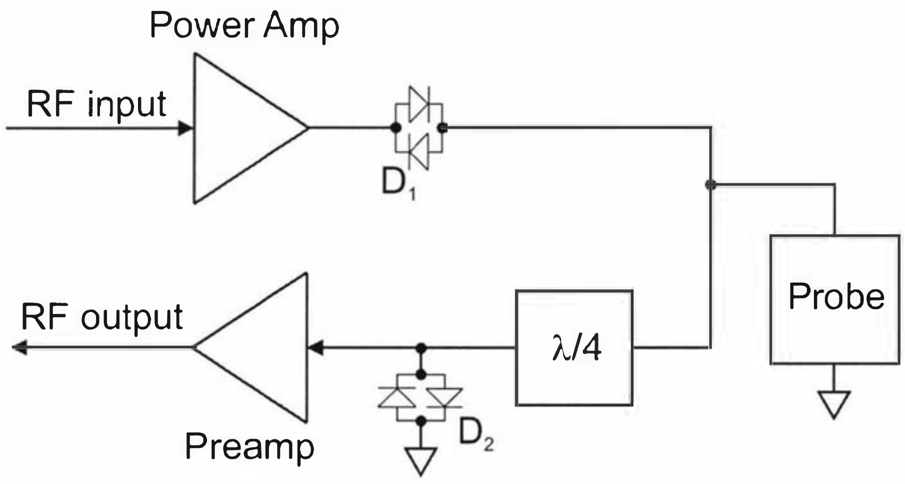


Figure 6.10 Quarter wave duplexer.

A schematic of the complete RF front end is shown in figure 6.11 where the quarter wave line has been implemented with a lumped element circuit consisting of an inductor (L1) and two equal capacitors (C1 and C2) [114]. When $\omega^2 LC = 1$, the lumped element circuit is at resonance and behaves as a quarter wave line with a characteristic impedance of $\frac{1}{\omega C}$. For a 3.3 MHz design, $C = C_1 = C_2 = 965\text{pF}$ and $L = L_1 = 2.41\,\mu\text{H}$.

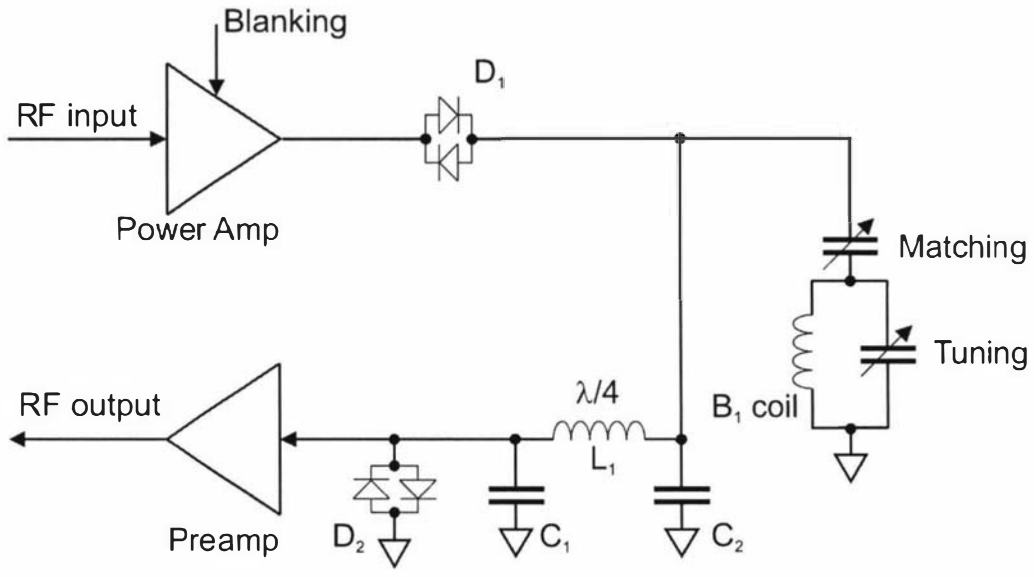


Figure 6.11 NMR-MOLE RF front end block diagram.

Since the RF front end was only constructed for the trialling of the new NMR-MOLE probe, it was unnecessary to include circuitry for the tuning and matching of the probe. The probe can be easily tuned and matched using a spectrum analyser that has a RF sweep output capability and a directional coupler like the ZFDC-10-6 from Minicircuits. The preamplifier and duplexer were placed in a metal enclosure which can be seen in figure 6.12.

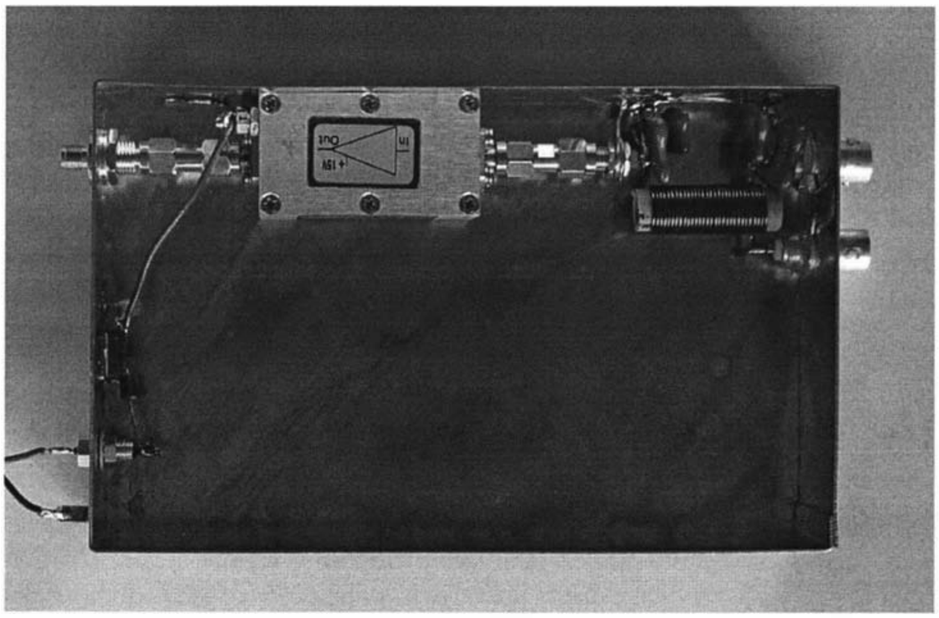


Figure 6.12 NMR-MOLE RF front end preamp/duplexer unit.

7.0 System integration and testing

7.1 Introduction

The final part of the NMR system development was the assembling of all the hardware and software and then testing it. Before all the hardware could be combined, a suitable enclosure needed to be considered as well as the interconnection of the various modules and the power supply. The aim was to produce a battery powered system that could be easily transported and used by a single user. It was desirable to have a flexible system that could support a range of probes and plug in modules, however, it was also desirable to produce a more compact unit specifically for the NMR-MOUSE. Therefore two different versions of the system were implemented both using the same core electronic modules. The flexible modular system nicknamed "Modspec" was enclosed in a standard readily available enclosure and allows NMR developers to change RF front ends to accommodate different probes. The more compact system was nicknamed "Lapspec" and this time a custom enclosure was constructed that was about the size of a laptop computer, hence the name. Lapspec is aimed for the non-NMR person who is only interested in obtaining information about materials.

The completed Modspec system was initially tested using a NMR-MOUSE and the RF front end described in section 6.3. This was not ideal as the RF front end modules were basically standalone units that could not easily be incorporated into either Modspec or Lapspec. One option was to design a custom RF front end for the NMR-MOUSE. However, after consultation with the mouse developers, it was decided to work together to reengineer their designs to fit onto the same sized circuit boards as used for all the other modules. The new preamp/duplexer board is shown in figure 7.0 and the RF power amplifier is shown in figure 7.1. The design of the new RF power amplifier was a considerable challenge as it was difficult to arrange the high power components within the very limited space.

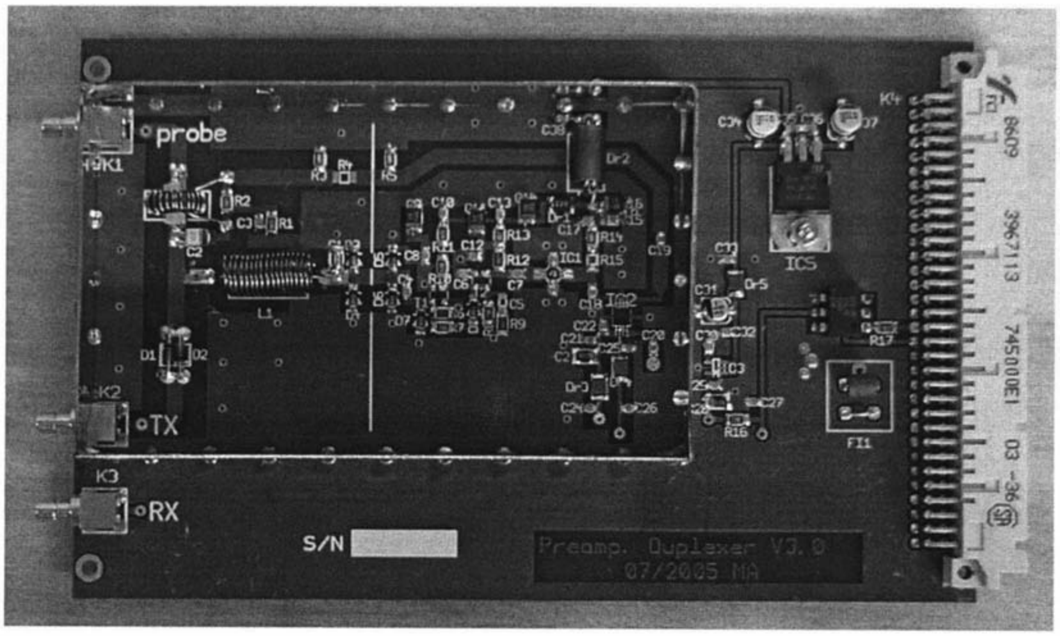


Figure 7.0 New 100 mm x 160 mm NMR-MOUSE Preamp/Duplexer module [22].

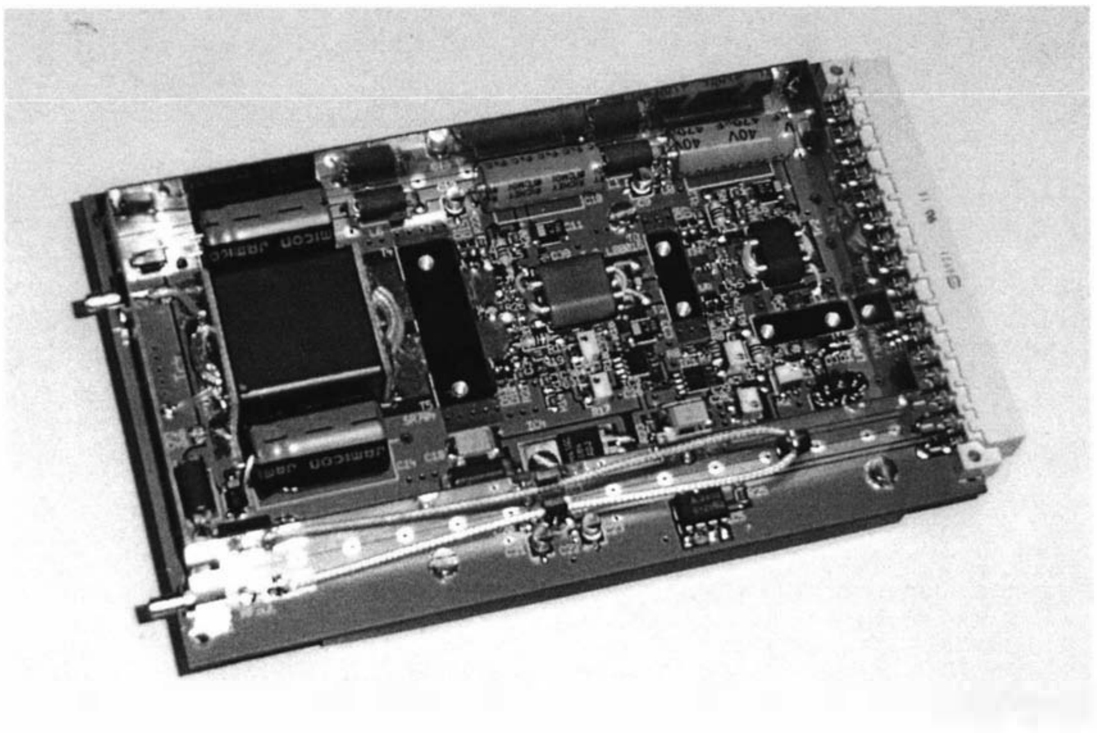


Figure 7.1 New 100 mm x 160 mm NMR-MOUSE 100 W RF power amplifier module [22].

The software also consists of various parts that need to be combined in a fashion to provide a range of solutions for different types of users. Again, like the hardware, a common platform is used that can be easily adapted to cater for various applications, users and systems. For some users and applications a "one button" user interface is desired where the experiments and data processing is all done automatically giving a simple pass/fail result. On the other hand, NMR developers need to be able to control every aspect of the system and so tools need to be provided to enable this. This flexibility has been provided by combining the user interface, scripting and data processing capabilities of Prospa together with the fully programmable DSP contained within the DSP/USB board.

7.2 Hardware integration.

7.2.1 Modspec

Modspec is the more flexible and modular version of the two systems as all the circuit boards were designed to fit into a standard Eurocard case. Such a case was purchased from Schroff [115] and is shown in figure 7.2. It is a 3 U high, 160 mm card depth, ¾ (of 19" rack) width, bench top enclosure with internal rails and PCB guides to support a backplane and PCB modules. The front extrusion and all the panels are aluminium and therefore provide screening as well as robustness.

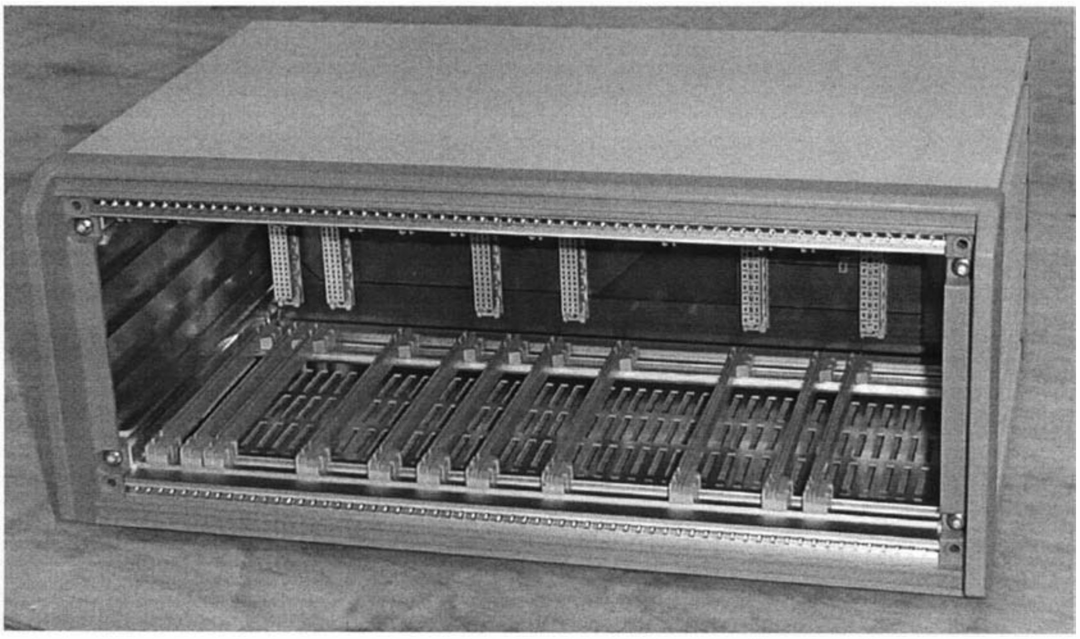


Figure 7.2 Modspec enclosure.

The interconnection between the modules is provided by a backplane board as shown in figure 7.3. The full schematic for the backplane is contained with appendix A5.

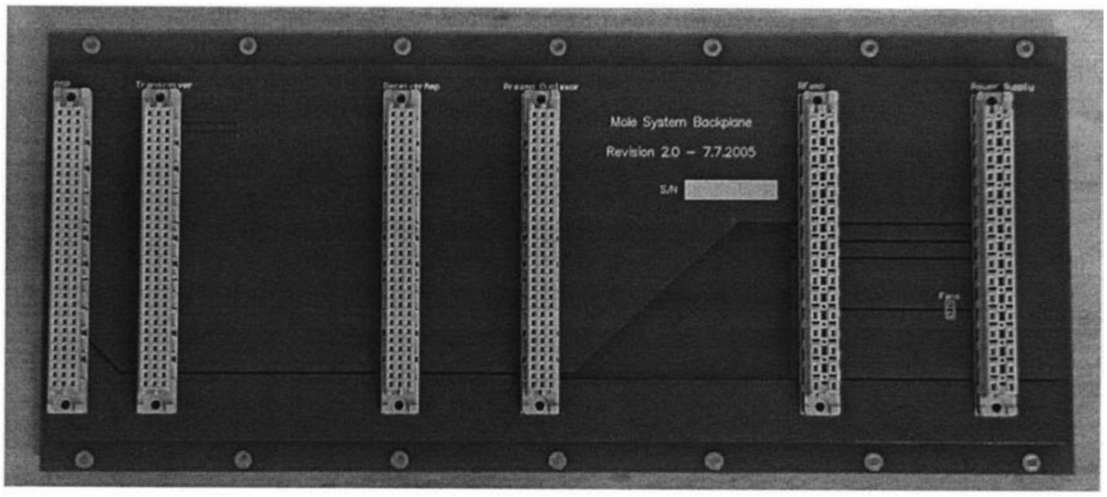


Figure 7.3 Modspec backplane.

Modspec as well as Lapspec are both designed to operate from a single 24 V DC power supply. This is to allow the use of batteries to power the system. The electronics modules, however, operate with various voltages and so some power conversion is necessary. Firstly, a 7 V rail is needed for the DSP/USB board, digital transceiver board and the receiver amplifier board. Initially 8 V was specified for these boards, but this was later reduced to 7 V to reduce power dissipation. The preamp/duplexer as well as the RF power amplifier have their own on-board voltage regulators and therefore can be connected directly to the 24 V rail. The ±15 V rails have been provided in Modspec to accommodate a planned additional gradient board.

The power supply was built on another 100 mm x 160 mm PCB so that it would slot into the Eurocard enclosure and connect to the backplane board. The schematic is shown in figure 7.4 and is also included within appendix A4.

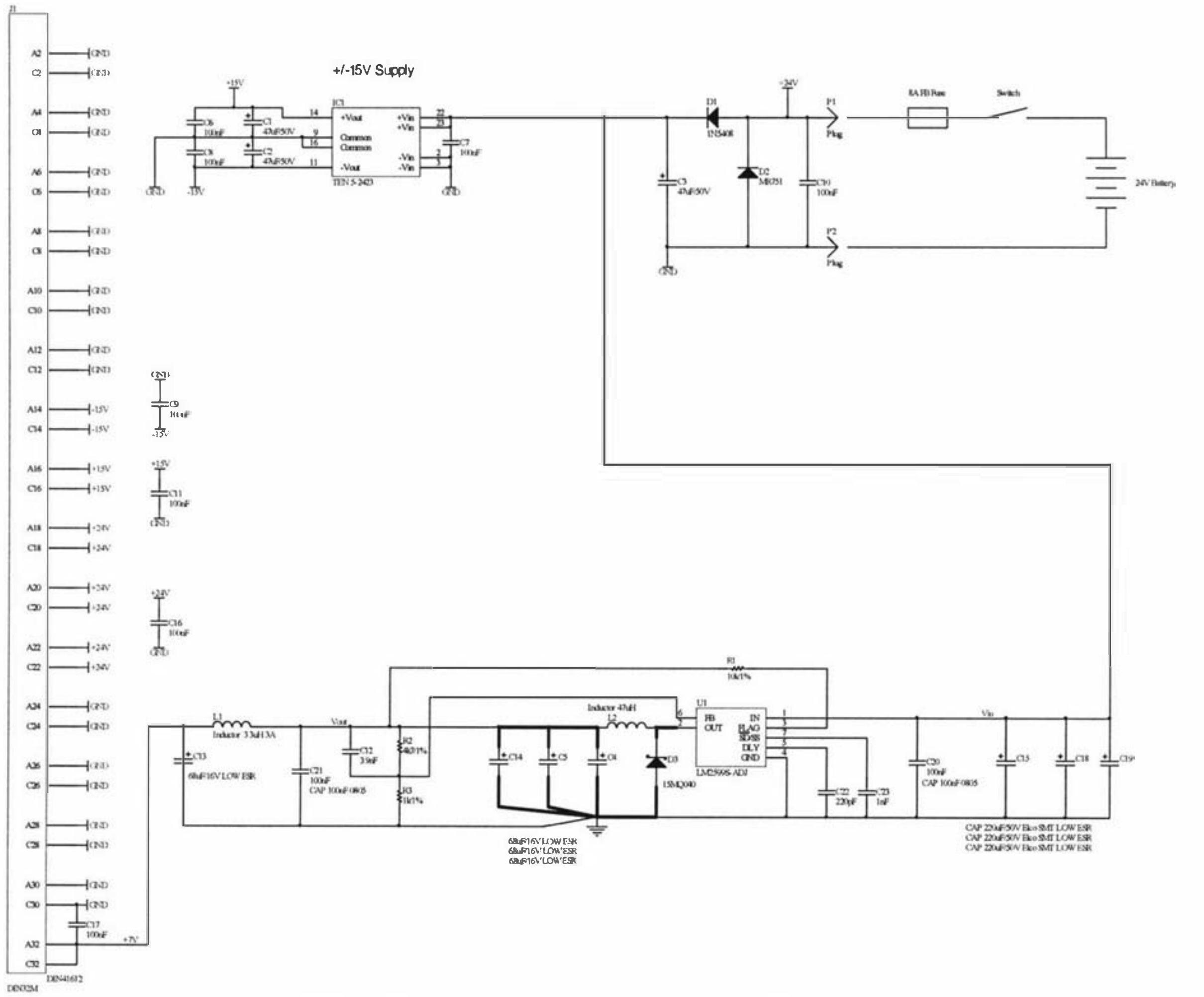


Figure 7.4 Modspec power supply schematic.

The system user provides the 24 V power feed from either batteries or a mains powered supply. As a precaution some polarity protection is included and is necessary when using batteries as they can source large currents which can produce disastrous results. Here a large diode (D2) is placed across the input so that if the wrong polarity is connected the diode will conduct and blow the fuse. Another diode (D1) is placed in series with the feed going to the voltage regulators to prevent a negative potential appearing across the inputs of the regulators. The ±15 V rails are generated using a 6 W DC-DC converter from Traco [116] that is capable of providing 200 mA at each of the two outputs. The 7 V rail is produced using an LM2599S "simple switcher" design from National Semiconductor [117] and can provide up to 3 A.

Using switch mode power supplies is not ideal as they can potentially produce interference due to the switching processes. The LM2599 device operates at 150 kHz and the Traco regulators at about 300 kHz so they should not be too much of a problem, but nevertheless extra screening and ground planes were included to minimise interference. The power supply board is also located within the enclosure as far away as possible from the sensitive modules. The completed 2 layer PCB excluding the Traco regulator is shown in figure 7.5. At this stage the ±15 V rails were not needed and so the components were not inserted.

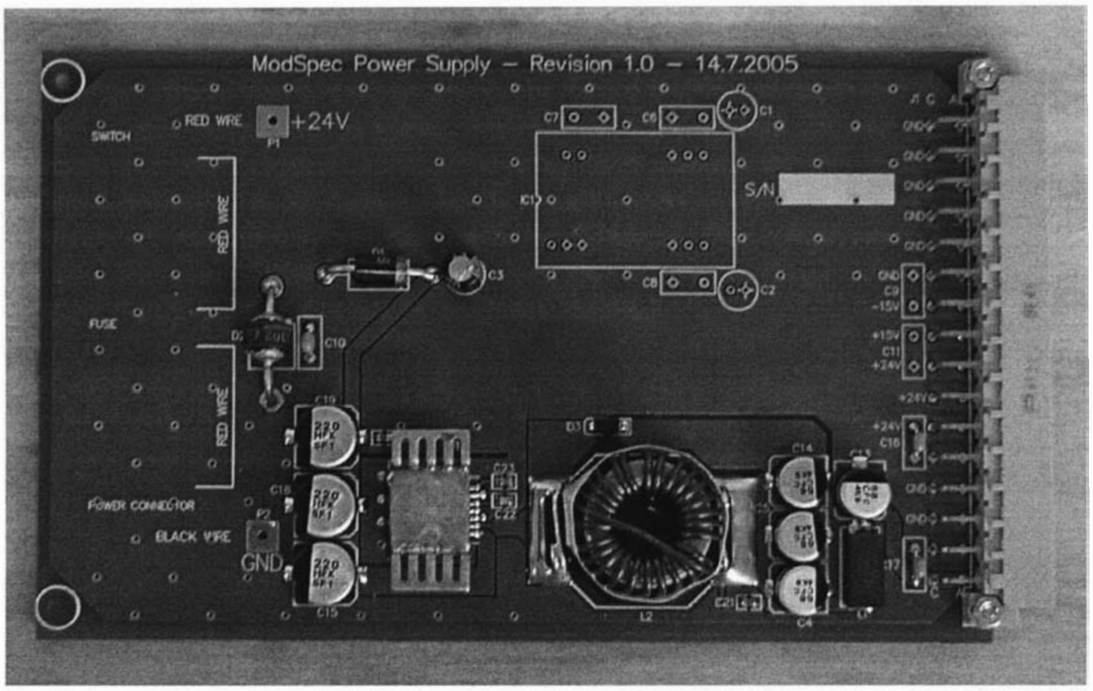


Figure 7.5 Modspec power supply module.

Together, the DSP/USB, digital transceiver and receiver amplifier boards require about 1.2 A continuously. This is well within the capabilities of the 7 V power supply. The main power consuming module within the system is the RF power amplifier. However this is only during the transmit phase of an experiment as all the other time the RF power stages are shut down. The RF power amplifier typically operates in class AB mode and is about 33% efficient [111]. This means that for a 250 W amplifier the peak power demand is on the order of 750 Watts. For a 24 V power supply this translates to more than 30 A. Fortunately the RF transmit pulses are usually very short ($< 10\ \mu s$) and so storage capacitors can be used to provide the peak currents. These capacitors are included within the RF power amplifier module as they need to be very close to the final RF power output stage. The inefficiency of the RF power amplifier could result in the dissipation of considerable amounts of heat and necessitate the use of large heatsinks. The total system power demand and the RF power amplifier power dissipation depend on the particular application.

The enclosure, together with modules, screening, and cooling fans is shown in figure 7.6. Extra space has been provided between the digital transceiver and the receiver amplifier to allow for a future gradient control board. The screens are connected to the chassis Earth and consist of some 1.5 mm thick aluminium sheets that slide into guide rails. The RF power amplifier and the digital transceiver each have a fan located beneath them to assist with cooling. Two 40 x 40 x 10 mm 12 V fans are connected in series with the 24 V power supply. A large amount of space has been allocated for the RF power amplifier heatsink and additional fans can be inserted if necessary. Each module has an individual front panel that has been laser cut out of 2.5 mm thick aluminium. These can be seen, together with the interconnecting cables and connectors, in figure 7.7.

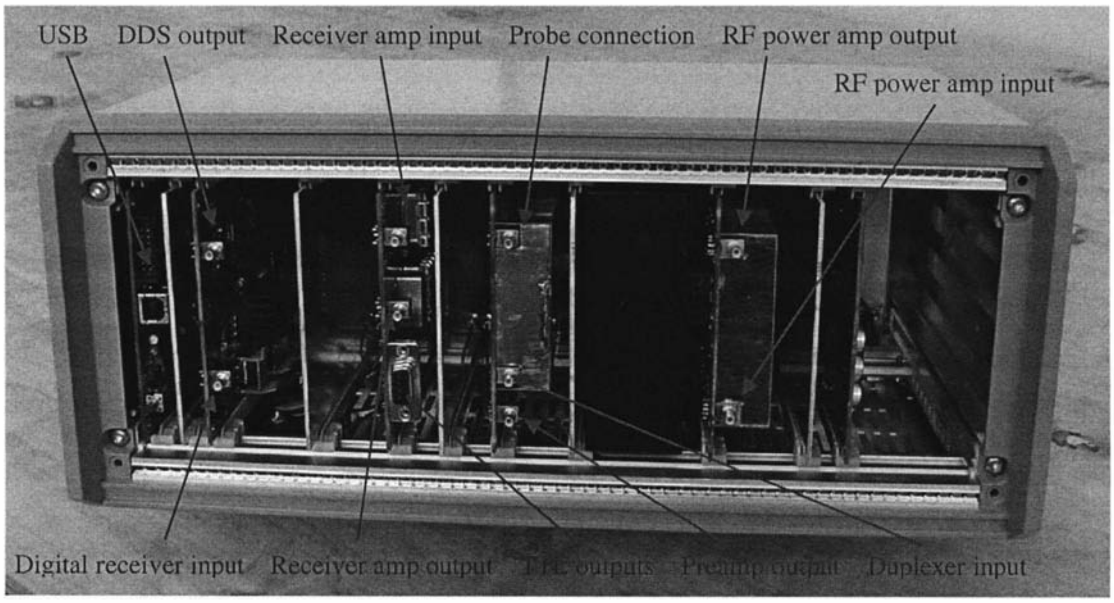


Figure 7.6 Modspec enclosure with modules, fans and screens.

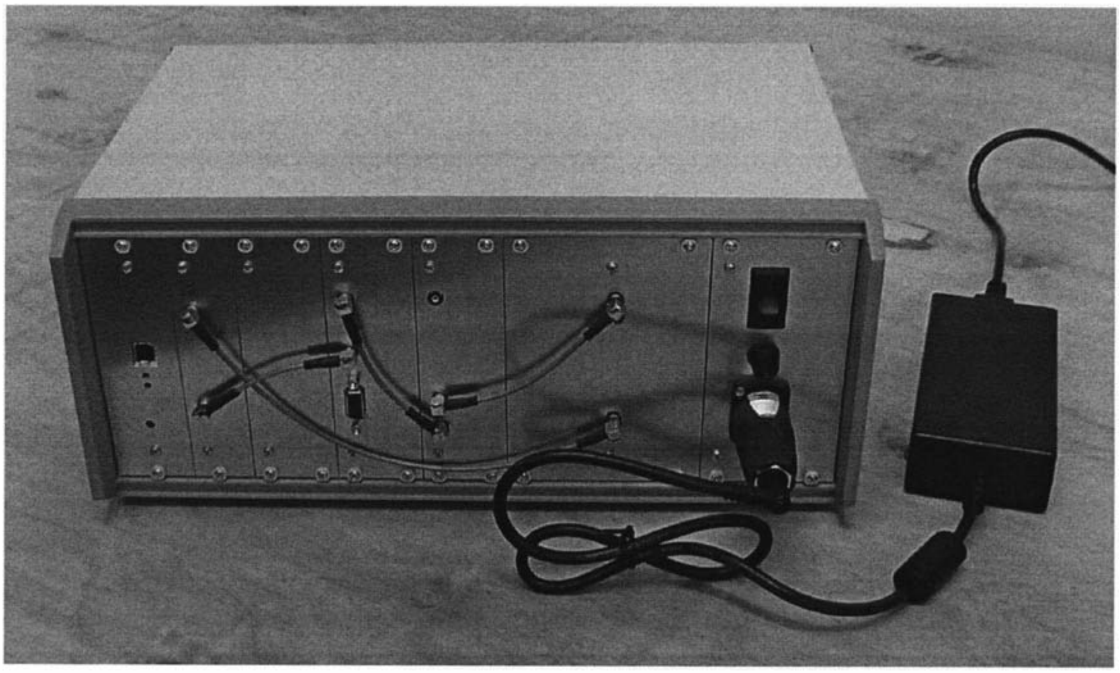


Figure 7.7 Fully assembled Modspec with front panels, cables and a 100 W, 24 V power supply.

7.2.2 Lapspec

Lapspec is a version that has been customised for the NMR-MOUSE. It still uses the same modules as Modspec but it is packaged into a smaller specially designed enclosure that does not allow for any future developments such as gradient control. The enclosure design is based around three identical 103 x 160 x 53 mm extruded aluminium enclosures placed side by side. These enclosures were obtained from Hammond Manufacturing [118] and are designed to house standard 100 x 160 mm eurocards. An exploded view of the complete enclosure design is shown in figure 7.8. A PCB backplane as well as a series of other aluminium panels provide the mechanical support resulting in a single rigid enclosure.

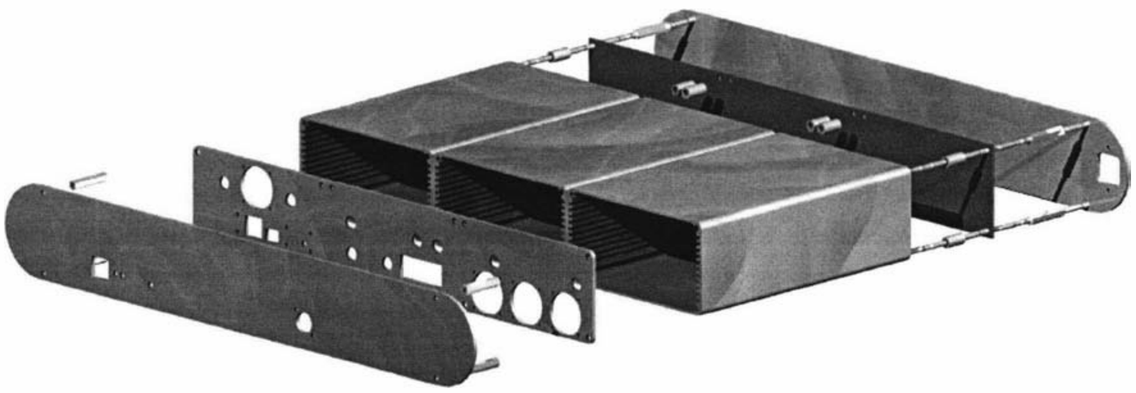


Figure 7.8 Lapspec enclosure design.

Again a backplane PCB provides the interconnection between the modules. Lapspec did not require the ±15 V power supply rails and so the 7 V power supply was included on the backplane and can be seen in figure 7.9. The Lapspec backplane schematic is included within appendix A6.

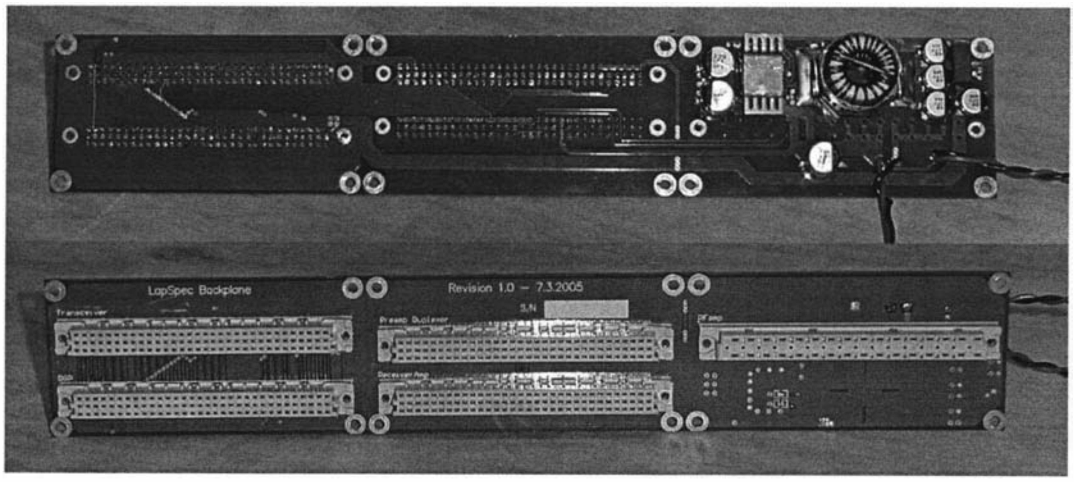


Figure 7.9 Lapspec backplane, including 7 V power supply (front and back views).

The aluminium enclosures together with some 1.5 mm thick aluminium sheets provide screening between the modules as shown in figure 7.10.

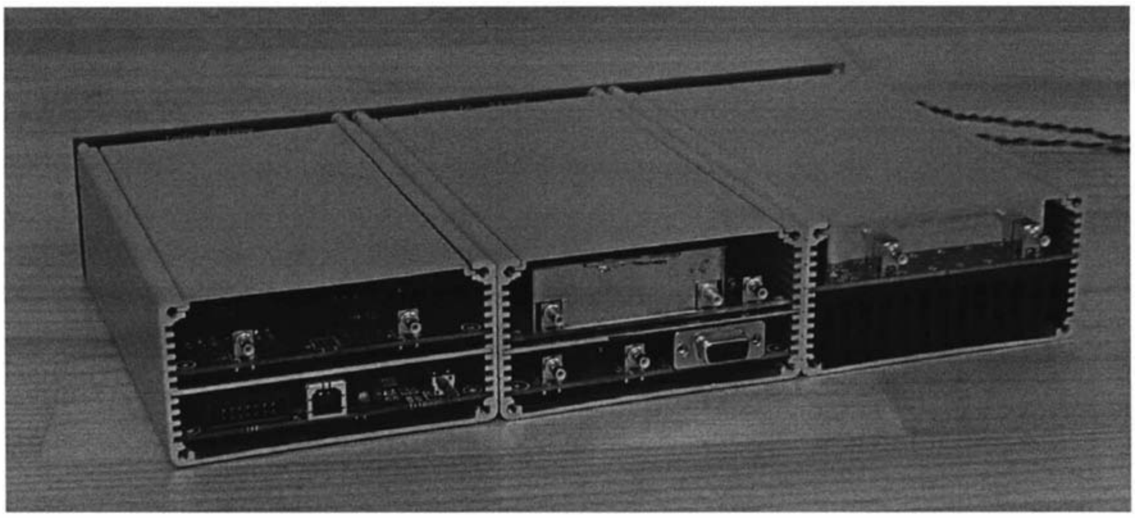


Figure 7.10 Lapspec with modules and screening.

An inner 2.5 mm thick laser cut aluminium front panel is then attached to the front of the internal enclosures to provide mechanical support and additional screening which can be seen in figure 7.11. Fans are also mounted onto this panel to provide cooling for the digital transceiver and the RF power amplifier. The RF interconnections between the modules are made using coaxial cabling and SMB connectors.

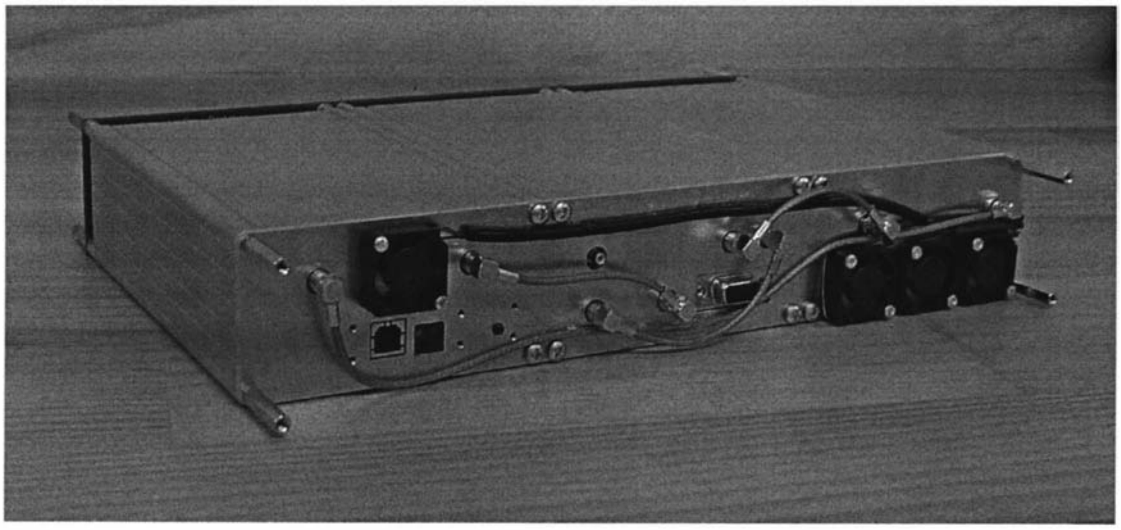


Figure 7.11 Lapspec with inner front panel, fans and cabling.

Front and back panels, also made out of 2.5 mm laser cut aluminium, are then attached as shown in figure 7.12 and the final enclosure is shown in figure 7.13 where an anodised aluminium shell provides the outer casing. The only connections available to the user are the 24 V DC power supply connector at the rear and the USB and probe connections at the front.

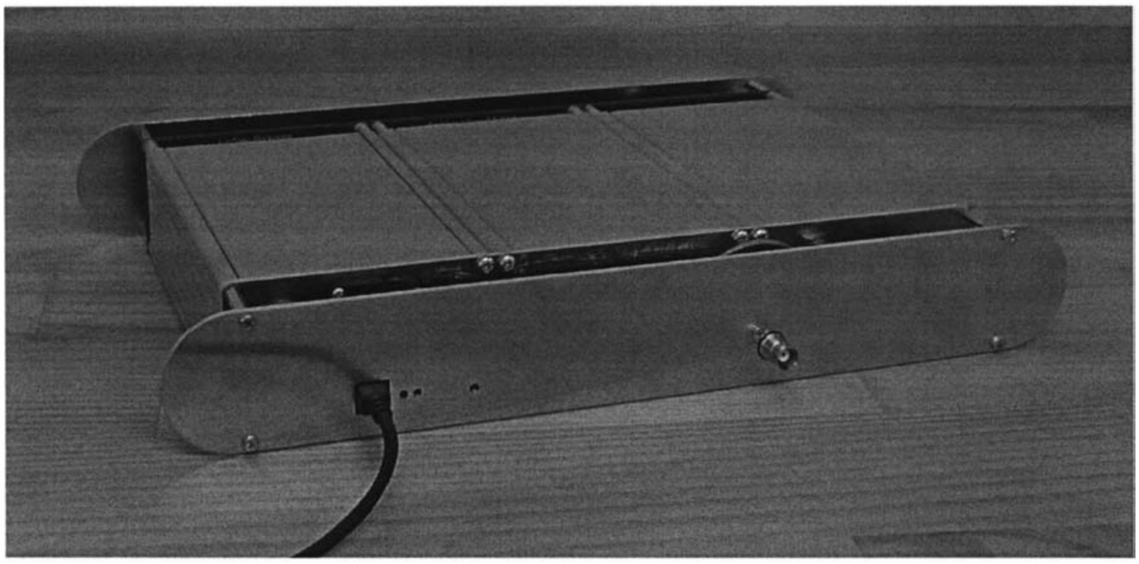


Figure 7.12 Lapspec with inner front panel, fans and cabling.

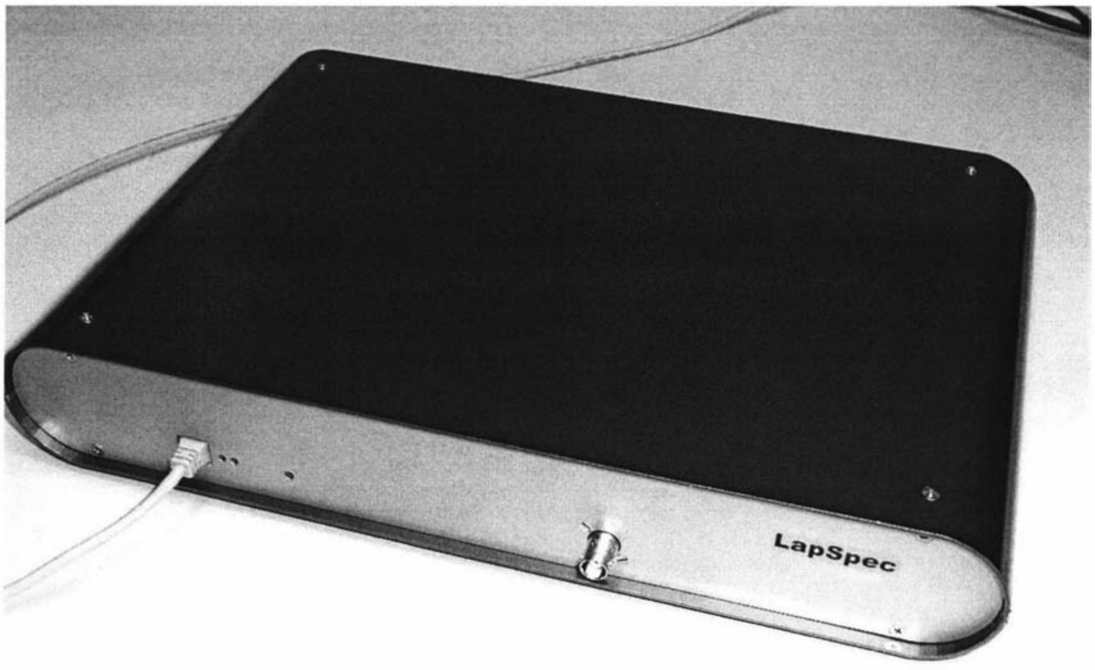


Figure 7.13 Complete Lapspec, measuring 360 x 240 x 55 mm and weighing 3.6 kg.

7.3 Software integration.

Like the hardware, the core of the software is the same for Modspec and Lapspec. The system has been designed to cater for a wide range of users. Modspec is the general purpose NMR development platform that is intended for those who are knowledgeable about NMR, whereas Lapspec is intended for the non-NMR user. The software, therefore, needs to be able to support a range of users without compromising flexibility and performance. The types of users can be divided into a series of levels as follows:

- Non-NMR user, can operate using just one button.
- Standard NMR user, can access a set of provided control panels and pulse sequences.
- NMR developer, can design control panels, macros and high level pulse programs.
- NMR expert, can develop pulse programs with "C" or Assembler for the DSP and new functions using the Prospa DLL interface.

Each of these different types of users are easily accommodated by simply building layers of software on top of each other and then restricting the user's access to the various layers. In the case of the user interface this can be implemented using the Prospa macro capabilities and in the case of the DSP/USB system this can be implemented by providing various pulse program development tools. Restricting users is important as the more privileges a user has the more damage they can do. Therefore the aim is to always try and keep users at the highest possible level. At this stage in the development of Modspec and Lapsec only the NMR developer and NMR expert levels are considered. Once the systems have been in operation for some time and some applications have become clear, the higher level layers can be developed.

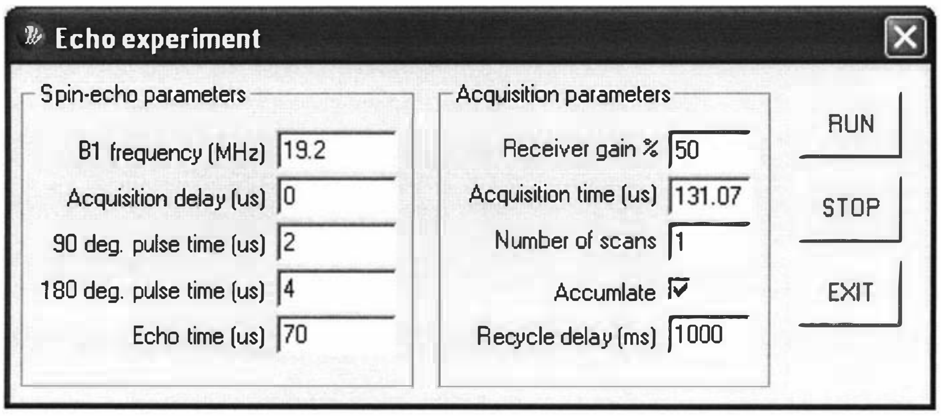
The software integration consists of interfacing two parts. The Prospa software and the DSP pulse programs. The data transfer between the two parts was described in detail in section 4.3.2 where a virtual link between Prospa and the DSP/USB system was formed using a DLL that contained "MyAPI". However, a protocol is still required to define the method of passing parameters from Prospa to the DSP/USB system. The idea here is that Prospa sends a preassembled pulse program, together with some parameters, to the DSP/USB system and then requests that it execute it. Experiments such as a T_2 experiment would be implemented by getting the DSP/USB system to execute a spin echo pulse program multiple times but with different parameters. These parameters would be sent in between each individual execution. The way that was chosen to implement this is to develop a Prospa macro and a DSP pulse program for each type of NMR experiment. This is now demonstrated using the example of a spin echo experiment. The complete listing of the spin-echo experiment macro is contained within appendix B2.

7.3.1 Prospa experiment macros

The experiment macro is first defined as a procedure so that it can be called by the Prospa runtime environment. The first part of the macro defines some constants that act as default parameters within the experiment. The graphical user interface is then defined as well as the functions of the user interface buttons. Prospa has a graphical tool for designing user interfaces which automatically generates code that can be simply inserted into the experiment macro.

```
procedure(echo)

# Initilaize value for parameters

   acqTime       = 131.072
   b1freq        = 19.2
   spare         = 0
   nrScans       = 1
   time90        = 2
   time180       = 4
   echoTime      = 70
   recycDelay    = 1000
   recgain       = 50

   accum         = "yes"

   n = window("Echo experiment", 37, 485, 464, 201)
      windowvar(timeLeft,timeRight)
      timeLeft     = 0
      timeRight    = acqTime
      edittext(3, 327, 55, 42)
      statictext(4, 228, 58, "left", "Acquisition time (us)")
      edittext(7, 132, 55, 61)
      statictext(8, 27, 60, "left", "Acquisition delay")
      edittext(9, 327, 78, 42)
      statictext(10, 241, 81, "left", "Number of scans")
      button(11, 392, 14, 52, 33, "RUN",
             enablecontrol(0,11,"false");
             enablecontrol(0,13,"false");
             echo:runpp();
             enablecontrol(0,11,"true");
             enablecontrol(0,15,"true");
             enablecontrol(0,13,"true");)
      button(13, 392, 98, 52, 33, "EXIT",
             closewindow(0);)
      groupbox(14, "Spin-echo parameters", 5, 8, 197, 154)
      button(15, 392, 56, 52, 33, "STOP",
             enablecontrol(0,11,"true");
             enablecontrol(0,13,"true");
             enablecontrol(0,15,"true");)
      checkbox(25, 327, 106, "no,yes", "yes")
      statictext(26, 271, 107, "left", "Accumlate")
      edittext(27, 132, 78, 61)
      statictext(28, 20, 83, "left", "90 deg. pulse time (us)")
      edittext(1, 327, 32, 42)
      statictext(6, 245, 35, "left", "Receiver gain %")
      groupbox(33, "Acquisition parameters", 213, 8, 166, 154)
      edittext(34, 132, 32, 61)
      statictext(35, 32, 37, "left", "B1 frequency (MHz)")
      edittext(5, 132, 101, 61)
      statictext(12, 14, 106, "left", "180 deg. pulse time (us)")
      edittext(16, 132, 124, 61)
      statictext(18, 59, 129, "left", "Echo time (us)")
      edittext(17, 327, 124, 42)
      statictext(2, 232, 129, "left", "Recycle delay (ms)")
      setpar(0,15,"mode","panic");

   showobjects(n)

# Initialise pulse program parameters

   setpar(0,3,"text",acqTime)
   setpar(0,1,"text",recgain)
   setpar(0,7,"text",spare)
   setpar(0,9,"text",nrScans)
   setpar(0,16,"text",echoTime)
   setpar(0,17,"text",recycDelay)
   setpar(0,34,"text",b1freq)
   setpar(0,27,"text",time90)
   setpar(0,5,"text",time180)

endproc()
```

The previous macro code generates the graphical window shown in figure 7.14.


Echo experiment

Spin-echo parameters
B1 frequency (MHz) 19.2
Acquisition delay (us) 0
90 deg. pulse time (us) 2
180 deg. pulse time (us) 4
Echo time (us) 70

Acquisition parameters
Receiver gain % 50
Acquisition time (us) 131.07
Number of scans 1
Accumlate ✓
Recycle delay (ms) 1000

Figure 7.14 Spin echo control panel.

When the run button is selected, the “runpp” procedure within the macro is then executed. It first starts by reading a preassembled pulse program (“echo.p”) from the computer’s hard-drive, it then downloads it into the DSP’s program memory starting at address 1000.

```
procedure(runpp)

  # Load pulse program and send to DSP

   cd("$appdir$\PulseSequences\Portable NMR\echo")
   pp = dspsrec("echo.p")
   dspwrite("p",0x1000,pp)
```

The parameters are downloaded once they have been extracted from the graphical user interface. The parameters are put into a format that is recognised by the DSP code and are placed into an array that is mapped to the internal memory within the DSP system.

```
# Extract pp parameters from GUI

   recgain      = getpar(0,1,"value")
   acqTime      = getpar(0,3,"value")
   spare        = getpar(0,7,"value")
   b1freq       = getpar(0,34,"value")
   time90       = getpar(0,27,"value")
   time180      = getpar(0,5,"value")
   echoTime     = getpar(0,16,"value")
   nrScans      = getpar(0,9,"value")
   recycDelay   = getpar(0,17,"value")

   nrPnts       = acqTime/0.512
   timeRight    = acqTime

   9852Fword = (( b1freq * 2^32 )/125)

   6620Fword = 9852Fword * 2

   echoTime = echoTime - (time90+time180)/2

   gamma = 2.69279e+08

  # Save parameters in an array (mapped to start of DSP "x" mem)

   p = matrix(32,1)

   p[0] = 0                                #9852 NCO phase etc
   p[1] = (9852Fword & 0xFF000000)/(2^24) #9852 NCO frequency
   p[2] = (9852Fword & 0xFF0000)/(2^16)
   p[3] = (9852Fword & 0xFF00)/(2^8)
   p[4] = (9852Fword & 0xFF)
   p[5] = (6620Fword & 0xFF000000)/(2^24) #6620 NCO frequency
   p[6] = (6620Fword & 0xFF0000)/(2^16)
   p[7] = (6620Fword & 0xFF00)/(2^8)
   p[8] = (6620Fword & 0xFF)
   p[9] = 0 #6620 NCO Phase
   p[10] = 0 #6620 NCO phase
   p[11] = nrPnts                #Number of data points
   p[12] = time90 * 50           #90 Pulse time
   p[13] = time180 * 50          #180 Pulse time
   p[14] = echoTime * 50         #Echo time
   p[15] = recgain * 1.5         #Receiver gain
   p[16] = recycDelay * 1000         #Recycle delay

  # Send parameters to DSP

   dspwritepar("x",0x00,p)
```

The macro then sets up the plot windows for the displaying of the data that will be retrieved later.

```
# Get display parameters from GUI

   accumulate = getpar(0,25,"text");

  # Set number of plot windows

   multiplot("1d",2,1)
  # Define matricies for display

   t = [0:1:nrPnts-1]/nrPnts*acqTime;
   f = ([0:1:nrPnts-1]-nrPnts/2)/acqTime;
```

The macro now sets up some memory buffers and then signals the DSP using the "dsprunpp" command to execute the downloaded pulse program. During pulse program execution Prospa uses a thread to poll the DSP/USB system (using USB interrupts) to see when it is finished. The user control panel is still active and so the user can abort the pulse program which results in an error being returned and the subsequent aborting of the pulse program execution macro. This would also occur if there was a transmission error. For normal operation the pulse program will signal when it is finished and so "dsprunpp" would return without an error.

```
# Run pulse program and accumlate data

   nrdataPnts = 2*nrPnts
   A = matrix(nrPnts)
   B = matrix(nrPnts)
   E = matrix(nrPnts)
   F = matrix(nrPnts)
   D = matrix(nrdataPnts)
   C = matrix(nrPnts)

   for(n = 1 to nrScans) # Scans loop

      r = dsprunpp() # Run pulse program
      if(r = 1 | r = 2)
        return # Abort if there's a problem
      endif
```

Upon completion of the pulse program execution, the data is retrieved from the DSP/USB board's "y" data memory starting at address 10000. The data is complex and is interleaved.

```
D = dspread("y",0x10000,nrdataPnts)
```

After retrieval the data is processed and displayed. The execution of pulse programs, the data retrieval, the processing and displaying of data can all be put within a loop so that multiple experiment executions can be performed. This is often done to implement signal averaging.

```
for(r = 0 to (nrPnts-2))
            A[r] = D[(2*r)]
            B[r] = D[((2*r)+ 1)]
      next(r)

      if(accumulate = "yes") # Get data
      E = E + A
      F = F + B
      C = sqrt(E^2 + F^2)
      Else
      E = A
      F = B
      C = sqrt(A^2 + B^2)
      endif

        drawplot("false")

         curplot("1d",1,1) # Display time domain data
         plot(t,E+i*F)
         #plot(t,C)
         title("text","FID data (Acc: $n$)","size",10)
         xlabel("text","time (us)","size",10)
         ylabel("text","Signal ","size",10)
         axes("fontsize",9)
         zoom1d(timeLeft,timeRight)

         curplot("1d",2,1) # Display frequency domain data
         s = mag(ft(E))
         plot(f,s)
         title("text","Magnitude spectrum (Acc: $n$)","size",10)
         xlabel("text","Frequency (kHz)","size",10)
         ylabel("text","","size",10)
         axes("fontsize",9)

      drawplot("true")

   next(n)

endproc()
```

To implement more complicated experiments such as T_2 or T_1 an additional loop would be used that would vary the parameters between individual or groups of pulse program executions. Only the parameters would need to be downloaded each time as the pulse program code would remain intact. If high speed loops are required then this could be done within the pulse program. As a rule, the DSP should only be used to implement the time critical part of the experiment. Prospa and the DSP/USB system operate as a master/slave arrangement with Prospa being the master.

7.3.2 Pulse programs

The pulse program that gets executed on the DSP/USB board basically generates analogue and digital signals to control the stimulation of the NMR system under investigation and subsequently acquire the corresponding response. The analogue signal is produced by the AD9852 DDS and the digital signals are 24 TTL outputs that are mapped to a single address (x:$20000) within the DSP/USB system's address space. The TTL output function allocations are defined as per table 7.0.

Data Bit	Bus Name	Allocated function
D0		Spare
D1		AD9852 update buffer enable (1 for enable)
D2		AD9852 update buffer and ADG901 Tx enable (0 for enable)
D3		ADG901 Tx enable (1 for Transmit)
D4		Reset AD6620 (0 for reset)
D5		Update AD6620 (1 for update)
D6		Update AD9852 (1 for update)
D7		Reset AD9852 (1 for reset)
D8	TTL0	TTL6 and TTL7 RF power amp enable (0 for enable)
D9	TTL1	Spare
D10	TTL2	Spare
D11	TTL3	Spare
D12	TTL4	Duplexer tune/receive (1 for tune)
D13	TTL5	Spare for duplexer
D14	TTL6	RF power amp enable (1 for enable)
D15	TTL7	Spare for RF power amp
D16		Spare on front Panel (External RF amp enable, 1 for enable)
D17		Spare on front Panel
D18		Spare on front Panel
D19		Spare on front Panel
D20		Spare on front Panel
D21		Spare on front Panel
D22		Spare on front Panel
D23		Spare on front Panel

Table 7.0 24 bit TTL output function allocations.

The NMR response is acquired using the AD6644 ADC and AD6620 digital receiver processor and is stored within the DSP/USB system's memory. The actual pulse program machine code for the DSP can be generated by different methods. The most basic method is to write the code in assembly language. At present, this is the preferred method as it allows complete control of the DSP and is optimal in terms of speed and code size. However, it is difficult to write as it requires a good knowledge of the DSP instruction set. Another option is to use a C compiler, but here there is a risk that the generated code may not execute fast enough for critical events. In addition, C can not cope with multiple memory spaces and so parts of the code would still have to be written in assembler. This combination of C and assembler will probably be the best solution in the interim until a dedicated pulse program compiler is produced. Parts of an assembly language pulse program used to implement the spin echo experiment are now shown as an example. The full listing can be found in appendix B1.

As with the experiment macros, the pulse programs also follow a standard form. The pulse program must first set up the memory spaces within the DSP/USB system and the electronic devices before the actual pulse sequence is executed. The parameters within the Prospa macro are loaded into the start of the DSP/USB system's x: memory. This space is defined at the beginning of the pulse program where each 24-bit word directly maps to the parameter data structure within the Prospa macro.

```
;*******************************************************************

        org     x:$0

PARAM_BASE      equu    *       ; Memory for pulse program parameters
TXW0            ds      1       ; Tx word 0
TXW1            ds      1       ; Tx word 1
TXW2            ds      1       ; Tx word 2
TXW3            ds      1       ; Tx word 3
TXW4            ds      1       ; Tx word 4
RXFW0           ds      1       ; Rx Frequency word 0
RXFW1           ds      1       ; Rx Frequency word 1
RXFW2           ds      1       ; Rx Frequency word 2
RXFW3           ds      1       ; Rx Frequency word 3
RXPW0           ds      1       ; Rx Phase word 0
RXPW1           ds      1       ; Rx Phase word 1
Ndatapoints     ds      1       ; Number of data points to acquire
T90             ds      1       ; 90 Pulse time
T180            ds      1       ; 180 Pulse time
Techo           ds      1       ; Echo time
Grec            ds      1       ; Receiver gain
Tcycle          ds      1       ; Recycle delay

;*******************************************************************
```

The code part of the DSP pulse program starts with synchronising itself to the 12.5MHz system heartbeat. This is explained in section 5.2.3.

```
      movep   #$0F3FE1,x:A_BCR      ;Set up wait states, 1 for AA3
      movep   #0,x:A_TLR0           ;Set up synch timer
      movep   #0,x:A_TCSR0          ;Disable timer
      movep   #$200261,x:A_TCSR0    ;Set up timer0 input capture
      move    #$80,a1
synch move    a1,x:$20000           ;Wait for synch
      jclr    #21,x:A_TCSR0,synch
      nop
```

The AD9852 DDS and AD6620 digital receive processor are then initialised through their 8-bit microprocessor interfaces. These interfaces can not operate at the full I/O speed of the DSP and so wait states need to be inserted within the DSP read/write cycles.

```
      movep   #$0FFFE1,x:A_BCR      ;Set up wait states, 7 for AA3
      nop
```

Following this, the AD9852 is initialised to the frequency defined in the parameter table and a default phase of 0. The DDS does not start until it receives an external synch pulse.

```
; Setup AD9852
```

```
        move    #$80,a1                 ;Reset 9852
        move    a1,x:$20000
        bsr     swait                   ;wait 100ns
        move    #$00,a1
        move    a1,x:$20000
        move    #$14,a1                 ;Set up control regs
        move    a1,x:$2009D
        move    #$24,a1
        move    a1,x:$2009E
        move    #$00,a1
        move    a1,x:$2009F
        move    #$60,a1
        move    a1,x:$200A0
        move    x:TXW1,a1                ;Output W1 to 9852
        move    a1,x:$20084
        move    x:TXW2,a1                ;Output W2 to 9852
        move    a1,x:$20085
        move    x:TXW3,a1                ;Output W3 to 9852
        move    a1,x:$20086
        move    x:TXW4,a1                ;Output W4 to 9852
        move    a1,x:$20087
        move    #$06,a1                  ;Set output amplitude 000-FFF
        move    a1,x:$200A1
        move    #$FF,a1
        move    a1,x:$200A2
```

The AD6620 requires a lot more to set up. The filter coefficients are loaded, the filter memory spaces are zeroed to remove any residual data and the control registers are set. The NCO frequency is set to the frequency defined in the parameter table. Again the AD6620 does not start until it receives an external synch pulse.

```
; Setup AD6620
;
        move    #$00,a1                 ;Reset 6620
        move    a1,x:$20000
        move    #$10,a1
        move    a1,x:$20000
        move    #$03,a1                 ;Put into soft reset
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        move    #$01,a1
        move    a1,x:$200C0
        move    #drecf,r1               ;Set up filter coeffs
        move    #$80,a1
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        do      #12,fcl                 ;12 coeffs
        move    p:(r1),a1
        lsr     #16,a
        move    a1,x:$200C2
        move    p:(r1),a1
        lsr     #8,a
        move    a1,x:$200C1
        move    p:(r1)+,a1
        move    a1,x:$200C0
fcl
        move    #$81,a1                 ;Clear filter data
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        do      #256,fdl
        move    #$00,a1
        move    a1,x:$200C4
        move    a1,x:$200C3
        move    a1,x:$200C2
        move    a1,x:$200C1
        move    a1,x:$200C0
fdl
        move    #$83,a1             ;Set up 6620 config regs
        move    a1,x:$200C7
        move    #$01,a1
        move    a1,x:$200C6
        move    #$00,a1             ;NCO control reg
        move    a1,x:$200C0
        move    #$ff,a1             ;NCO SYNC control reg
        move    a1,x:$200C3
        move    #$ff,a1
        move    a1,x:$200C2
        move    #$ff,a1
        move    a1,x:$200C1
        move    #$ff,a1
        move    a1,x:$200C0
        move    x:RXFW0,x0          ;NCO FREQ control reg
        move    x0,x:$200C3
        move    x:RXFW1,x0
        move    x0,x:$200C2
        move    x:RXFW2,x0
        move    x0,x:$200C1
        move    x:RXFW3,x0
        move    x0,x:$200C0
        move    #$00,a1             ;NCO PHASE control reg
        move    a1,x:$200C1
        move    #$00,a1
        move    a1,x:$200C0
        move    #$02,a1             ;INPUT/CIC2 scale reg
        move    a1,x:$200C0
        move    #$03,a1             ;Mcic2-1 reg
        move    a1,x:$200C0
        move    #$0A,a1             ;CIC5 scale reg
        move    a1,x:$200C0
        move    #$07,a1             ;Mcic5-1 reg
        move    a1,x:$200C0
        move    #$04,a1             ;OUTPUT/RCF control reg
        move    a1,x:$200C0
        move    #$00,a1             ;Mrcf-1 reg
        move    a1,x:$200C0
        move    #$00,a1             ;RCF ADDRESS OFFSET reg
        move    a1,x:$200C0
        move    #11,a1              ;Ntaps-1 reg
        move    a1,x:$200C0
        move    #$00,a1             ;Reserved reg
        move    a1,x:$200C0
        move    #$03,a1             ;MODE CONTROL reg
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        move    #$00,a1
        move    a1,x:$200C0
```

The AD6620 and AD9852 are then synchronised.

```
        move    #$12,a1             ;Turn on 9852 synch buffer
        move    a1,x:$20000
        bsr     swait
        move    #$72,a1             ;Toggle SYNCs
        move    a1,x:$20000
        bsr     swait
        move    #$12,a1
        move    a1,x:$20000
```

The gain of the receiver amplifier is then set to the value defined in the control panel. The two AD8369 amplifier devices are interfaced to one of the DSP's synchronous serial ports which, after the gain is set, is shut down to reduce system noise.

```
; Setup Receiver gain
      movep #$2C,x:A_PCRC            ;Set up SSI 0
      movep #$10080A,x:A_CRA0
      movep #$13C3C,x:A_CRB0
      move  x:Grec,a1          ;receiver gain 0000-FFFF
      movep #$0000,x:A_PDRE    ;select first gain block
      movep a,x:A_TX00         ;Set up first gain block
      move  #10,r7
      bsr   wait
      movep #$0004,x:A_PDRE    ;select next gain block
      movep a,x:A_TX00         ;Set up second gain block
      move  #10,r7
      bsr   wait
      movep #$0007,x:A_PDRE    ;select unused
      movep #$3c3c,x:A_CRB0    ;Shut down SSI 0
      movep #$00,x:A_PCRC
```

After the initialisation of the hardware devices the program enters the main part. An internal DSP timer is used to implement an event system with a timing resolution of 20 ns. The timer counts down and when it reaches zero it sets a flag which signals for the next event. That event sets the time for the following event. An event counter is used to point to the next event. An initial delay of 1 μs is set before the first event is executed. This delay gives time for the analogue parts of the system to settle. Each event performs a specific task at an exact time. The delay times are usually determined by parameters within the parameter area and can easily be changed between experiments.

```
move    #$00,r1                    ;use r1 as the event counter
        movep   #0,x:A_TLR2        ;Set up event timer
        move    #50,r3
        movep   r3,x:A_TCPR2       ;Set for first event
        movep   #$A01,x:A_TCSR2

evenl   jclr    #21,x:A_TCSR2,evenl ;Wait for event timer flag
        move    (r1)+              ;Increment event counter
        move    r1,b
        cmp     #1,b
        beq     event1
        cmp     #2,b
        beq     event2
        cmp     #3,b
        beq     event3
        cmp     #4,b
        beq     event4
        cmp     #5,b
        beq     event5
        bra     event6

;***************************************************************
;       Event 1, Unblank RF amp
;
event1  move    #50,r3             ;1us delay
        movep   r3,x:A_TCPR2       ;Set for next event
        movep   #$200A01,x:A_TCSR2 ;Clear timer flag
        move    #$1C012,a1         ;Unblank RF amp
        move    a1,x:$20000
        bra     evenl
```

```
;
;**************************************************************
;       Event 2, Start TX 90 pulse
;
event2  move    x:T90,r3
        movep   r3,x:A_TCPR2        ;Set for next event
        movep   #$200A01,x:A_TCSR2  ;Clear timer flag
        move    #$1C01A,a1          ;start 90x Pulse
        move    a1,x:$20000
        bra     evenl
;
;**************************************************************
;       Event 3, Stop TX 90 Pulse, blank RF amp, change phase
;
event3  move    x:Techo,r3
        movep   r3,x:A_TCPR2        ;Set for next event
        movep   #$200A01,x:A_TCSR2  ;Clear timer flag
        move    #$12,a1             ;Stop 90 Pulse and blank RF amp
        move    a1,x:$20000
        move    #$10,a1             ;set phase of 9852 to y
        move    a1,x:$20080         ;0000 = 0, 1000 = 90
        move    #$00,a1             ;2000 = 180, 3000 = 270
        move    a1,x:$20081
        move    #$52,a1             ;Update phase by toggling 9852
        move    a1,x:$20000
        move    #$12,a1
        move    a1,x:$20000
        bra     evenl
;
;**************************************************************
;       Event 4, Unblank RF amp
;
event4  move    #50,r3              ;1us delay
        movep   r3,x:A_TCPR2        ;Set for next event
        movep   #$200A01,x:A_TCSR2  ;Clear timer flag
        move    #$1C012,a1          ;Unblank RF amp
        move    a1,x:$20000
        bra     evenl
;
;**************************************************************
;       Event 5, start TX 180 pulse
;
event5  move    x:T180,r3
        movep   r3,x:A_TCPR2          ;Set for next event
        movep   #$200A01,x:A_TCSR2    ;Clear timer flag
        move    #$1C01A,a1            ;start 180y Pulse
        move    a1,x:$20000
        bra     evenl
;
;**************************************************************
;       Event 6, Stop TX 180 Pulse and blank RF amp
;
event6  movep   #$200A00,x:A_TCSR2    ;Clear timer flag and stop
        move    #$12,a1               ;Stop 180y Pulse and blank RF
        move    a1,x:$20000
        bra     adcs
;
;**************************************************************
```

For a spin echo sequence the acquisition is the last event so in this case it follows on from event 6. The sample rate is determined by the AD6620 and the DSP's Timer1 input capture system is used to sense when data is available. A software loop then reads in the desired amount of data and stores it in the DSP/USB board's off chip Y memory starting at address $10000.

```
;Sample data

adcs    movep   #0,x:A_TLR1          ;Set up ADC timer
        movep   #0,x:A_TCSR1         ;Disable timer
        movep   #$361,x:A_TCSR1      ;Set up timer1 input capture
        move    #$10000,r5           ;Make r5 point to start of mem
        move    #$0400,r7            ;Load number of samples into r7
adcl    jclr    #21,x:A_TCSR1,adcl   ;Wait for timer1 flag
        movep   #$200361,x:A_TCSR1   ;Clear timer1 flag
        move    x:$20000,a1          ;Load data from channel A
        lsl     #8,a                 ;make it 24bit
        move    a1,y:(r5)+           ;Write to memory
        move    x:$20000,a1          ;Load data from channel B
        lsl     #8,a                 ;make it 24bit
        move    a1,y:(r5)+           ;Write to memory
        move    (r7)-                ;Decrement count
        move    r7,b
        tst     b
        beq     simp_e               ;Stop if =0
        bra     adcl                 ;Otherwise Loop again
simp_e
```

At the end of the pulse sequence the devices are put into a reset state.

```
        move    #$80,a1              ;Reset 6620,9852
        move    a1,x:$20000
```

Often this simple spin-echo sequence is repeated with different parameters to implement other experiments such as T_2 analysis. It is also possible within a pulse sequence to rapidly change the phase and amplitude of the DDS output allowing more complicated sequences such as the CPMG sequence to be easily developed. The phase of the AD6620 NCO can also be altered within a pulse sequence.

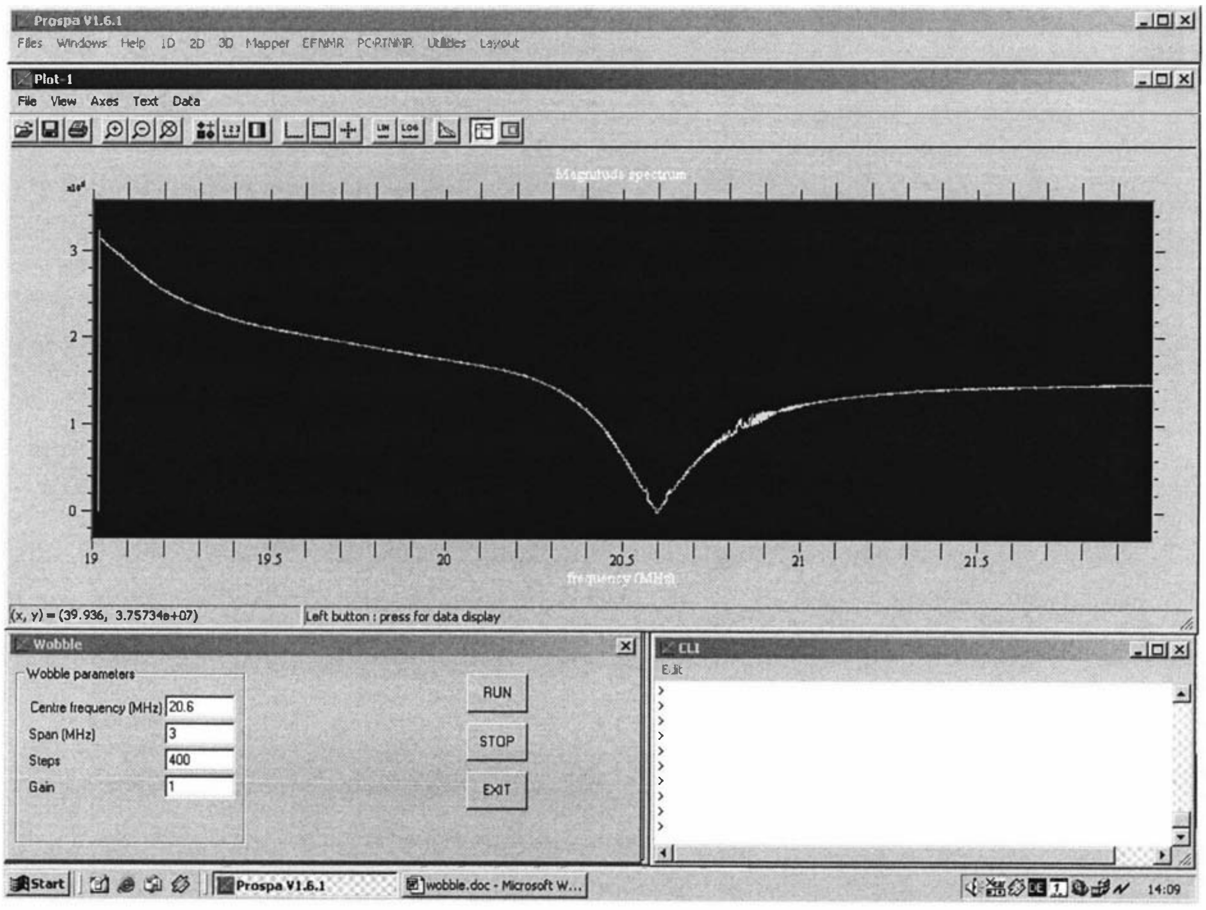
7.4 System testing

To test the system, a series of Prospa experiment macros and DSP pulse programs needed to be developed. These can be found on the CD at the back of this thesis. The first set of tests were done with the Modpsec system and the NMR-MOUSE.

7.4.1 Probe tuning and matching

Before an NMR experiment can be performed it always necessary to tune and match the probe to 50 Ω. The common method for this is to "wobble" the probe. Here the probe is driven with a low power swept frequency source while simultaneously monitoring the reflected power. The match to 50 Ω is indicated by a reflected power minimum. Impedance matching is important in order to maximise the power transfer from the RF amplifier to the probe and the probe to the preamplifier and also to ensure that the duplexer circuit works correctly.

Wobble is an important feature for portable NMR systems as one does not want to have to carry around a spectrum analyser or impedance analyser just to do this task. A wobble control panel and pulse program was implemented and a screen shot is shown in figure 7.15.



(x, y) = (39.936, 3.75734e+07)

Left button : press for data display

Wobble parameters

Centre frequency (MHz) 20.6
Span (MHz) 3
Steps 400
Gain 1

Figure 7.15 Wobble control panel and display. The horizontal position of the dip in the trace indicates the frequency where the probe is being matched. The level of the lowest point of the dip corresponds to the probe impedance. A zero level equates to 50 Ω. In this case the probe is matched to 50 Ω at 20.6 MHz.

The wobble experiment was also necessary for testing the tuning/matching hardware capabilities. The frequencies of the NCOs within the AD9852 and AD6620 were incremented in a stepped fashion and reflected power data was obtained at each of the steps using the digital receiver. Each sweep took about 2 ms for a span of 3 MHz covered in 400 discrete steps. However the screen update rate was much lower as it took about 200 ms for the data to be transferred and displayed on the screen. This resulted in a screen update rate of 5 per second which still gave a real time feel when adjusting the probe's capacitors and therefore made it quick and easy to tune and match the probe.

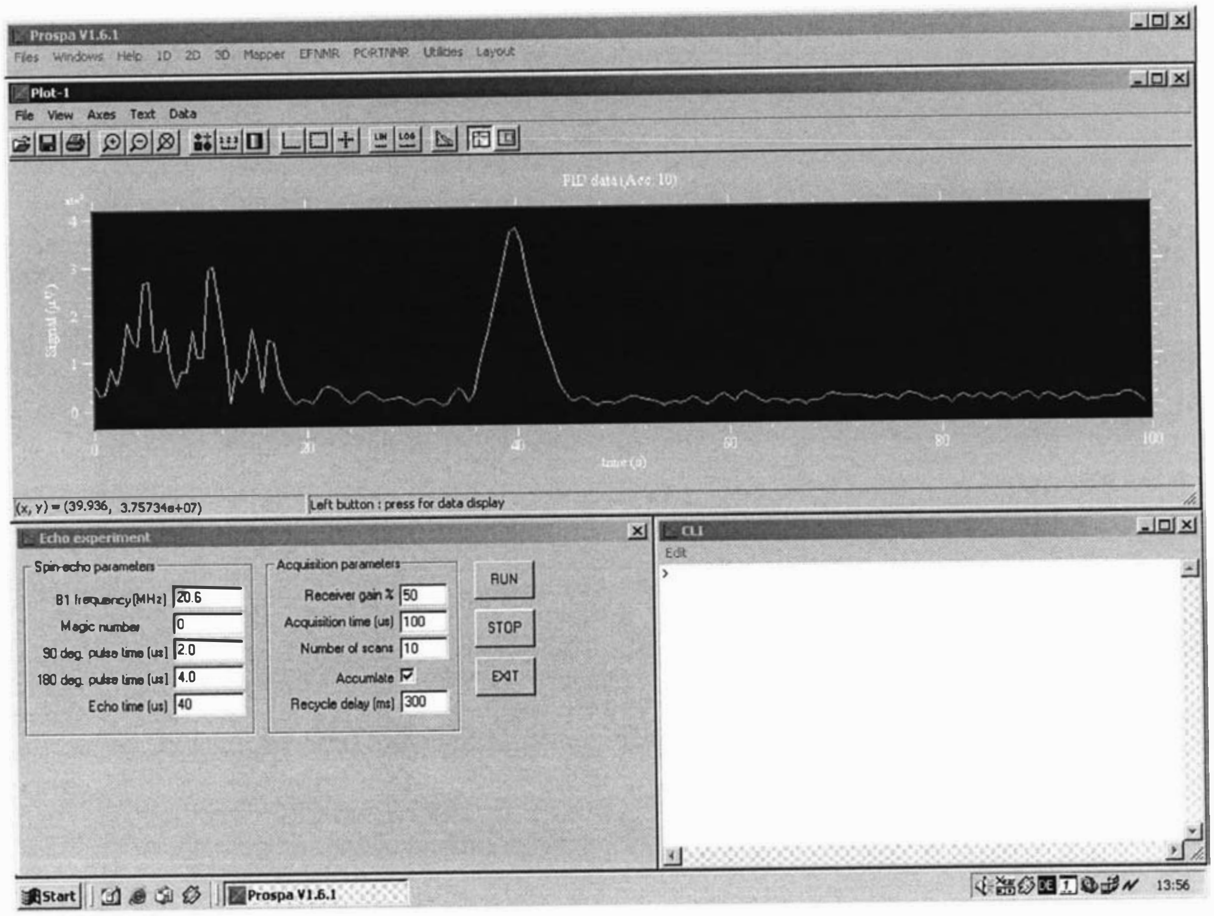
The tuning/matching capacitors are not entirely independent and each has hysteresis. These factors increase the difficulty of tuning and make it easy to overshoot the optimal setting. Increasing the screen update rate reduces the overshoot problem and therefore significantly speeds up the task. Built in tuning/matching is an important capability of this system as it successfully addresses an issue that has been, in the past, a real problem with many NMR systems. The probe was tuned to 20.6 MHz which corresponds to the dip in the sweep output, in figure 7.15

7.4.2 NMR-MOUSE spin-echo experiment

The critical test was the ability to obtain an NMR signal. The NMR-MOUSE and most other permanent magnet based surface probes have a relatively inhomogeneous sensitive region. Consequently there is little point in trying to obtain an FID as it will have decayed before the probe has finished ringing down. Therefore a spin-echo experiment was performed and the result is shown in figure 7.16.

The spin-echo sequence consists of a 90_x pulse followed by a 180_y pulse and the delay between them is the echo time which in this case was 40 μs. The rubber sample used was a standard test sample provided by the probe developers. The spin-echo sequence was repeated 10 times with signal averaging to demonstrate that the synchronisation was working correctly. Signal averaging was also needed due to the poor S/N performance of the probe. The noise at the beginning is a combination of probe ring down and receiver recovery. The spin-echo itself is very short, lasting only about 10 μs and this highlights how inhomogeneous the sample region is.

With this sequence the 180° pulse was produced by simply doubling the duration of the 90° pulse. However for inhomogeneous probes, it is more common to the double the RF power. This keeps the pulse durations as short as possible and therefore maximises the excited bandwidth and therefore the amount of signal received. The RF power level that was used was 50 W. It is difficult to judge the performance of the core portable NMR system in this way as the performance is greatly dependent upon the performance of the probe and preamp. However it was still a useful exercise as it showed that the system was functioning correctly. The result was presented to the probe manufacturer who was completely satisfied with the S/N performance, noting that it matched the best they had ever achieved using other spectrometers [22].



(x, y) = (39.936, 3.75734e+07)

Left button : press for data display

Spin-echo parameters

B1 frequency(MHz)	20.6
Magic number	0
90 deg. pulse time (us)	2.0
180 deg. pulse time (us)	4.0
Echo time (us)	40

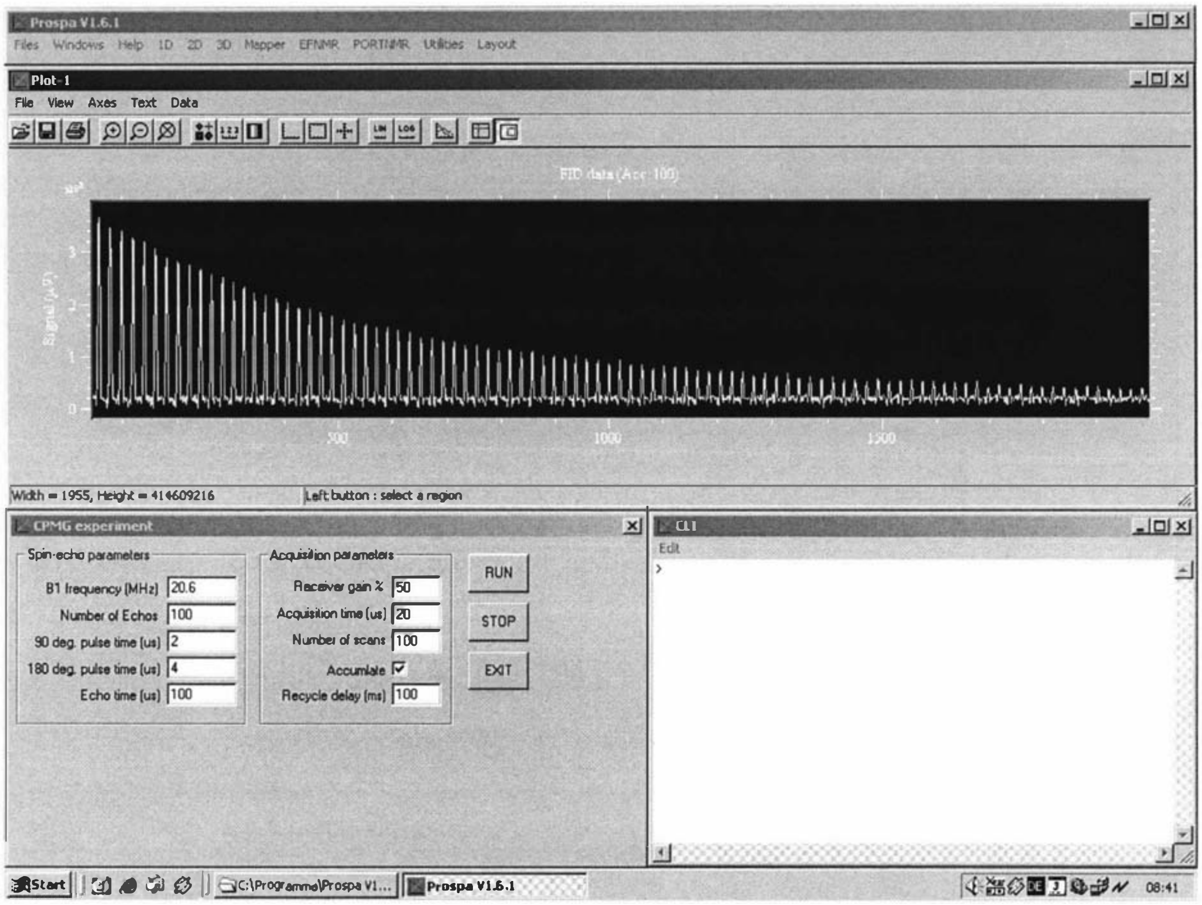
Acquisition parameters

Receiver gain %	50
Acquisition time (us)	100
Number of scans	10
Accumlate	✓
Recycle delay (ms)	300

Figure 7.16 Spin-Echo experiment with a rubber sample. 10 scans were performed and averaged with a recycle delay of 300 ms. The echo time used was 40 μs.

7.4.3 NMR-MOUSE CPMG experiment

The NMR-MOUSE is often used to determine the T_2 relaxation time of a sample. T_2 is related to the softness of a material which in turn is related to the inter-bonding of the individual molecules. The most efficient way of performing T_2 measurements is to use the CPMG technique. This is much like a standard spin-echo sequence but this time a whole series of 180° refocusing pulses are applied to generate a series of spin-echoes. The control panel and some data are shown in figure 7.17. Again the rubber sample was used but with 100 scans and an echo time of 100 μs. The horizontal axis displays the sample number which translates to a total time window of 20 ms.



Width = 1955, Height = 414609216

Left button : select a region

Spin-echo parameters

B1 frequency (MHz) 20.6
Number of Echos 100
90 deg. pulse time (us) 2
180 deg. pulse time (us) 4
Echo time (us) 100

Acquisition parameters

Receiver gain % 50
Acquisition time (us) 20
Number of scans 100
Accumulate ✓
Recycle delay (ms) 100

Figure 7.17 CPMG experiment with rubber sample.

The spin-echoes follow a single exponential decay which a time constant (T_2) of approximately 4×10^{-3} s which is typical for this type of sample [22]. This result was also shown to the probe manufacturer who accepted it as confirmation that the system was functioning correctly [22]. The CPMG experiment is a good experiment for testing the spectrometer as a CPMG experiment is sensitive to phase and timing errors. Any errors would result in irregularities in the spin-echo formation. This will appear as a faster than expected decay of the exponential envelope and as irregularities in the echo amplitudes.

7.4.4 NMR-MOLE spin-echo experiment

The experiments with the NMR-MOUSE demonstrated that the spectrometer and the software were functioning correctly so there was confidence that it could then be used with the new untried NMR-MOLE probe. The reason the probe had not yet been tested was that the commercial spectrometers could not go below 6 MHz. There are difficulties when working with new probes such as not knowing the exact field strength of the homogenous region or what RF power and pulse durations to use. The NMR-MOLE is designed to have a flat RF coil, but to help in determining the field strength and homogeneity a small coil was constructed of copper wire wound around a 10 mm NMR tube (figure 7.18). A solenoidal coil gave much greater S/N performance than a flat coil and therefore made it easier to find a signal. The tube was filled with water and then placed in the predicted region of best homogeneity which was centred 25 mm from the probe's surface. This region was previously measured with a Hall probe to have an equivalent field of 3.145 MHz. The coil was tuned and matched using a spectrum analyser and reflectance bridge. This interim technique was required as the RF front end for the NMR-MOLE lacked a directional coupler.

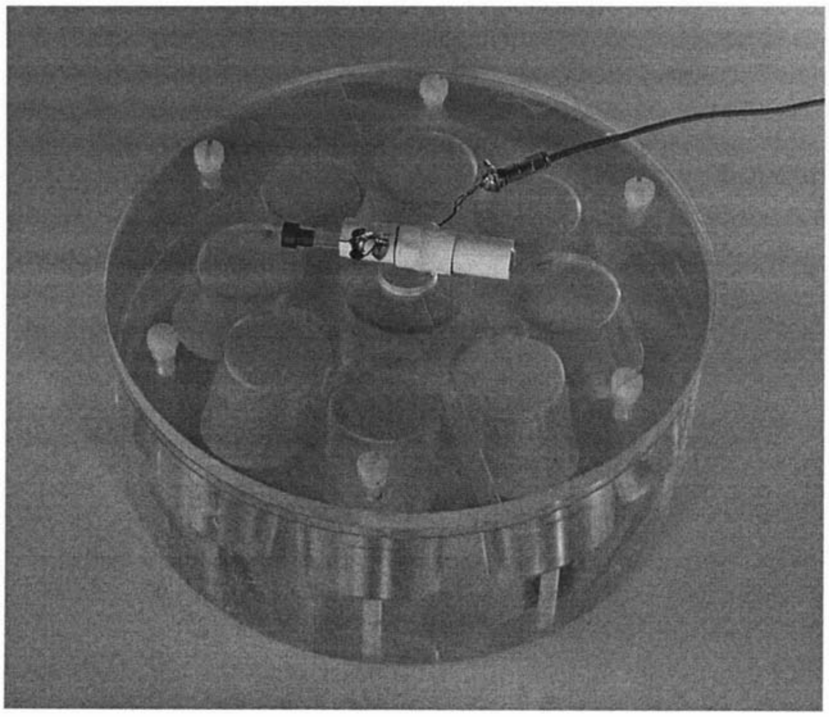


Figure 7.18 NMR-MOLE magnet array together with RF coil.

The spin-echo experiment was then used with an echo time of 100 μs to obtain the signal shown in figure 7.19. The RF excitation power was 1.5 W, the 90° pulse time was determined to be 5 μs and the 180 pulse time was 10 μs. The experiment was repeated fifty times with signal averaging and an inter-experiment delay of 1 s. The duration of the spin-echo is around 30 μs and this equates to a frequency deviation within the relatively homogeneous region of 30 kHz. This is rather broad but, now that it can be measured, it can used by the probe designer to optimise the probe. After obtaining this signal the Modspec system together with the NMR-MOLE probe was handed over to the probe designers so that they could start the refining process. The experiment with the NMR-MOLE probe demonstrated the capability of working at low frequencies and the suitability of the spectrometer for the intended application. This may be the end of the spectrometer development, but it is only the beginning of a whole new area of research in New Zealand.

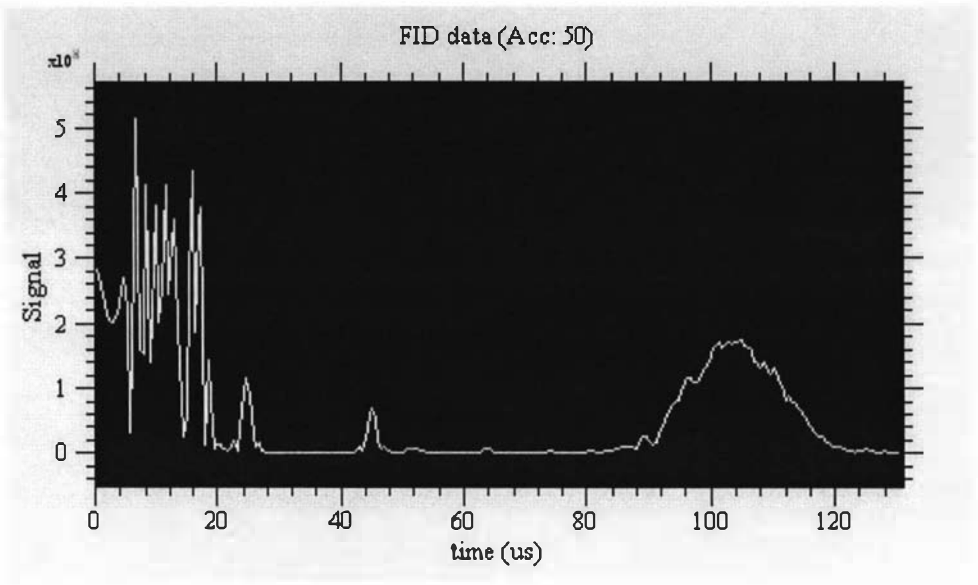


Figure 7.19 First spin echo from NMR-MOLE (magnitude plot).

8.0 Conclusions

With the introduction of Portable NMR technologies, measurements can now be performed on a whole host of materials that can not be taken into the lab. One example is the stones or bricks within historic structures. NMR measurements can be used to aid the development of suitable sealants and coatings to preserve these monuments of antiquity. The aim of this work was to further developments in portable NMR by providing a platform technology that others can build upon to develop solutions in a range of application areas. These areas include: sea ice research, timber properties of growing trees, monitoring the curing of the polymer resins used in boat building, testing of surface coatings and the measurement of moisture within soil, timber and concrete. This list is not complete and will continue to grow as more and more applications are considered.

At the beginning of this thesis it was described how a total solution is made up of various components. This was illustrated in figure 1.0 which corresponds to the upper section of figure 8.0. Each of the components was deemed to require specific skills. Consequently the development of a total solution needed a multidisciplinary approach. An essential component of the solution is the electronics section which has now been achieved as a result of this work. This was done by breaking the electronics section itself into three sub units as shown in figure 8.0. The DSP/USB system, the digital transceiver and the RF front end were treated as individual modules (building blocks) to simplify the design process as well as to provide flexibility to allow the system to be adapted to new applications. Being able to break down the design into smaller modules was very helpful as it essentially divided up the development into a set of smaller scale problems, each with its own set of criteria or performance requirements. This is very much like the top-down approach that is often used in software development.

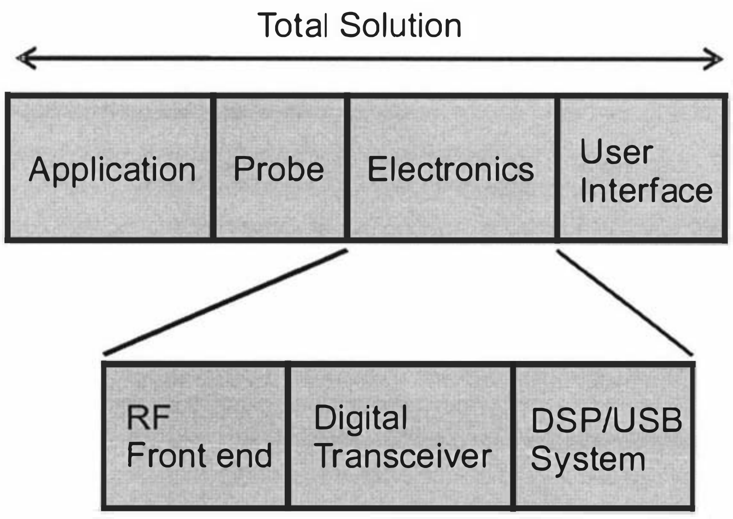


Figure 8.0 Electronic modules within the total solution.

This work has given the NMR community a new capability in the form of a compact mobile spectrometer platform that can be adapted to a range of probes. This should result in some new instruments for industry. These new instruments should be relatively inexpensive to produce, again unlocking new opportunities and hopefully making NMR a common industry technology. This potential has now been recognised by some NMR companies and has lead to an increase in activity in this area. The work described here has been a part of this new activity and a significant impact has been made. Lapspec was first made public at the 4th Mobile NMR conference [119] that was held in Aachen Germany. At that time it was only presented as a proof of concept system as it was not fully functioning. A working version of Lapspec was recently demonstrated at the 5th Mobile NMR conference [120] where it was the most compact and advanced spectrometer available. At this conference one company (Tecmag) showed a circuit board that they plan to use as the basis for a new mobile spectrometer called “LapNMR”. These developments are exciting as it means that the concept of portable NMR is now an established reality and an emerging area of research.

This work is still in progress, but now a whole team is working with Modspec or Lapspec as the core NMR instrumentation technology to produce the other parts of the total solution for a number of different applications. Lapspec and Modspec will, in the near future, go into production. A number of systems are already in use by other research groups, one in the United States of America and one in Germany. The designs are presently in the process of being handed over to the newly formed NMR company called “magritek” and will join the Earth’s field NMR product. Modspec has been relabelled as “Kea”, a native bird known for its intelligence and durability. The manufacturers of the NMR-MOUSE have arranged with magritek to market Lapspec together with their probes so that they can offer complete solutions. They also are making available a set of RF front end modules for the Kea system.

There is still a lot of work that could be done to optimise the Modspec and Lapspec systems and hence extract the last little bit of performance. This, like most projects, will continue in an incremental fashion. In chapter 1 it was said that NMR is technology driven. Therefore continual improvements and increases in capability are to be expected. One technology area that has advanced steadily is Field Programmable Gate Array logic (FPGA). It would now be possible to implement the digital receive processor part of the transceiver, with an operating sample rate of 100 MHz, using one of the new FPGA devices. Analogue Devices now provides a 14-bit ADC that can operate up to 105 MSPS, so a new higher speed digital receiver will shortly be investigated. Using a 100 MHz sample rate will simplify the synchronisation logic as it would be operating at the same rate as the DSP. The DDS could also be made to operate at 200 MHz giving an improvement in the transmit section capabilities as well as further simplifying the synchronisation circuit. Using an FPGA will provide the ability to gain access to the raw high-sample-rate data and therefore it would be possible to perform measurements on the quantisation noise. This feature could be used to assess the phase noise performance of the sample clock and would therefore assist in the development of low phase noise oscillators, which will be necessary if sub-sampling is implemented.

The system core part of the NMR system can and will be used for other applications as it is a general purpose control/acquisition system. Two applications that have also been considered are a scanning tunnelling microscope and a compression rheometer. For NMR applications some further work that could be undertaken is the development of a pulse program editor. This could be a graphical or a custom language editor that would not require the user to know anything about C or assembly language for the

DSP. The editor would take the user input and compile executable code for the DSP. The possibility of linking the PC programmers interface (MyAPI) into other applications such as Matlab is something that could also be explored.

It is envisaged that imaging will be implemented. This will place greater demands on the memory and data transfer medium. Therefore in the future it is most likely that the DSP/USB system board will be redesigned with increased memory and a higher speed data communications link. Wireless USB will be released this year and will be an attractive option as it will provide high throughput as well as wireless connectivity.

Some of the major limitations of the NMR spectrometer are due to the RF electronics. Therefore a new area of focus could be to make the RF electronics smaller and more efficient. The main issue being the power demands and thermal management of the RF power amplifier which place demands on the system power supply and enclosure size. For Lapspec it would be good to be able to provide a small lightweight, battery powered backpack system that uses a wireless link to a handheld Personal Digital Assistant (PDA).

Another area that could be thoroughly investigated is signal processing techniques. Often if you know what you are looking for then it is a lot easier to find it. This is the idea behind matched filters, cross correlation receivers and neural networks. For example in the case of the NMR-MOUSE probe, the shape (or envelope) of the spin-echo signal is known, and therefore filters can be optimized to match this characteristic and hence improve the S/N performance. Devices are now available with plenty of processing power to implement these types of features.

This completed work is really the initial phase of portable NMR development within New Zealand. To move on from here will require fanning out in many directions and will therefore require the involvement of other people with different backgrounds. This project was broad in its scope and encompassed both hardware and software. More specifically the skill areas that were needed to complete this work were, software engineering for both the DSP and Windows, high speed analogue and digital electronics design, digital signal processing techniques, RF engineering, mechanical design, project management, Protel CAD, thermal design, power electronics and NMR techniques.

This work has been successful and has resulted in a platform technology that will now assist in the development of application specific NMR systems. This will help to continue the task of taking NMR out of the lab and into the field and will lead to some more exciting new developments in the field of portable NMR.

References

[1] P. T. Callaghan, *Principals of Nuclear Magnetic Resonance Microscopy*, Oxford University Press, 1991.

[2] C. D. Eccles, C. J. Clark, S. L. Codd and R. Dykstra, Proceedings of the Fifth Electronics New Zealand Conference, p45-50, 1998.

[3] B. P. Hills, C. J. Clark, Ann Report NMR Spect, 50, p76-117, 2003.

[4] B. Blümich, S. Anferova, S. Sharma, A. L. Segre and C. Federici, J. Magn. Reson. 161, p204-209, 2003.

[5] N. Proietti, D. Capitani, E. Pedemonte, B. Blümich and A. L. Segre, J. Magn. Reson. 170, p113-120, 2004.

[6] S. Sharma, F. Casanova, W. Wache, A. Segre and B. Blümich, Magn. Reson. Imaging, 21, p249-255, 2003.

[7] P. J. McDonald, Prog. Nucl. Magn. Reson. Spect. in Physics, 30, p69, 1997

[8] G. Eidmann, R. Savelsberg, P. Blumler, and B. Blumich, J. Magn. Reson. A 122, p104-109, 1996.

[9] magritek Ltd, 32 Salamanca road, Kelburn, Wellington 6005, New Zealand, http://www.magritek.com

[10] R. Dykstra, *The Development of a Portable Earth's Field NMR System for the study of Antarctic Sea Ice*, Masters thesis, Massey University, 2001.

[11] C. D. Eccles, *Introduction to NMR imaging,* Physics, Massey University, 1999, Unpublished document.

[12] D. I. Hoult, Concepts in Magnetic Resonance, 12(4), 173, 2000.

[13] P. Edwards, Chemistry, Massey University, Private communication.

[14] E. L. Hahn, Phys. Rev. 80, p580, 1950.

[15] H. Y. Carr, E. M. Purcell, Phys. Rev. 94, p630, 1954.

[16] S. Meiboom, D. Gill, Rev. Sci. Instr. 29, p688, 1959.

[17] Bruker BioSpin GmbH, Silberstreifen 4, 76287 Rheinstetten, Germany, http://www.bruker-biospin.de

[18] Siemens AG, Wittelsbacherplatz 2, D-80333 Munich, Germany, http://www.medical.siemens.com

[19] Quantum Magnetics, Inc, 15175 Innovation Drive, San Diego, CA 92128, http://www.qm.com

[20] Tecmag Inc, 6006 Bellair Blvd, Houston, TX 77081, U.S.A, http://www.tecmag.com.

[21] Stelar s.r.l, 27035 Mede (PV), Italy, http://www.stelar.it

[22] M. Adams, RWTH, Aachen, Germany, Private communication.

[23] G. J. Bene, Abstracts of 4th International Symposium on Magnetic Resonance, Rehovoth, Israel, p25-27, August, 1971, (Weizmann Inst. Science, 1971).

[24] O. A. Shushakov, V. M. Fomenko, V. I. Yashchuk, A. S. Krivosheev, E. Fukushima, S. A. Altobelli and V. S. Kuskovsky, Waste Management (WM'04) Symposium, Tucson, Arizona, February 29-March 4, 2004.

[25] B. F. Melton, V. L. Pollak, Journal of Magnetic Resonance, A-122, 42, 1996.

[26] P. T. Callaghan, R. Dykstra, C. D. Eccles, T. G. Haskell, and J. D. Seymour, Cold Regions Science and Technology, 29, p153-171, 1999.

[27] O. Ripeka Mercier, M. W. Hunter and P.T. Callaghan, Cold Regions Science and Technology, 42, p96-105, 2005.

[28] J. Stepisnik, M. Kos, G. Planinsic and V. Erzen, Journal of Magnetic Resonance, A-107, p167, 1994.

[29] P. T. Callaghan, M. LeGros, Am. J. Phys., 50(8), p709, 1982.

[30] A. Mohoric, J. Stepisnik, M. Kos, and G. Planinsic, Journal of Magnetic Resonance, 136, p22, 1999.

[31] G. Planinsic, J. Stepisnik and M. Kos, Journal of Magnetic Resonance, A-110, p170, 1994.

[32] R. L. Kleinberg, Encyclopedia of Nuclear Magnetic Resonance, John Wiley & Sons, Ltd, p4960-4969, 1995.

[33] R. L. Kleinberg, J. A. Jackson, Concepts in Magnetic Resonance, 13(6), p340-342, 2001.

[34] F. Bloch, W. W. Hanson and M. Packard, Phys. Rev. 70, p474, 1946.

[35] R. J. S. Brown, Concepts in Magnetic Resonance, 13(6), p344-366, 2001.

[36] J. A. Jackson, Concepts in Magnetic Resonance, 13(6), p368-378, 2001.

[37] J. A. Jackson, L. B. Burnett and J. F. Harmon, Journal of Magnetic Resonance, 41, p411-421, 1980.

[38] M. N. Miller, Concepts in Magnetic Resonance, 13(6), p379-385, 2001.

[39] Z. Taicher, G. Coates, Y. Gitartz and L. Berman, Magnetic Resonance Imaging, 12(2), p285-289, 1994.

[40] R. L. Kleinberg, Concepts in Magnetic Resonance, 13(6), p396-403, 2001.

[41] D. Allen, et al, Oilfield Review, p2-13, Autumn, 2000.

[42] B. Blümich, V. Anferov, S. Anferova, M. Klein, R. Fechete, M. Adams and F. Casanova, Magn. Reson. Eng. 15 (4), p255-261, 2002.

[43] S. Anferova, V. Anferov, M. Adams, P. Blümler, K. Hailu, K. Kupferschläger, M. J. M. Mallett, G. Schroeder, S. Sharma and B. Blümich, Concept Magn. Reson. 15(1), p15-25, 2002.

[44] H. Kühn, M. Klein, A. Wiesmath, D. E. Demco, B. Blümich, J. Kelm and P. W. Gold, Magn. Res. Imaging, 19, p497-499, 2001.

[45] F. Balibanu, K. Hailu, R. Eymael, D.E. Demco and B. Blümich, J. Magn. Reson. 145, p246-258, 2000.

[46] M. Todica, R. Fechete and B. Blümich, J. Magn. Reson. 164, p220-227, 2003.

[47] B. Blümich, P. Blümler, G. Eidmann, A. Guthausen, R. Haken, U. Schmitz, K. Saito and G. Zimmer, Magn. Reson. Imaging, 16, p479-484, 1998.

[48] A. Guthausen, G. Zimmer, P. Blümler and B. Blümich, J. Magn. Reson. 130, p1-7, 1998.

[49] http://www.nmr-mouse.de/

[50] http://www.mpip-mainz.mpg.de/~bluemler/mouse/index.htm

[51] L. J. Parrot, Magazine of Concrete Research, 43, No. 154, p45-52, 1991.

[52] E. Fukushima, J. A. Jackson, United States Patent 6,828,892.

[53] M. W. Hunter, Physics, Victoria University of Wellington, New Zealand, Private communication.

[54] P. T. Callaghan, R. Dykstra, C. D. Eccles and M. W. Hunter, New Zealand patent application 520114 / International Patent Application No. PCT/ NZ03/00149

[55] H. Raich, P. Blumler, Concepts in Magnetic Resonance, Vol. 23B(1), p16-25, 2004.

[56] B. Parkinson, Physics, Victoria University of Wellington, New Zealand, Private communication.

[57] R. Dykstra, Proceedings of the Ninth Electronics New Zealand Conference, p6-11, 2002.

[58] The MathWorks, Inc, 3 Apple Hill Drive, Natick, MA 01760-2098, U.S.A, http://www.mathworks.com

[59] National Instruments Corporation, 11500 N Mopac Expwy, Austin, TX 78759-3504, U.S.A, http://www.ni.com

[60] Apple Computer, 1 Infinite Loop, Cupertino, CA 95014, U.S.A, http://www.apple.com.

[61] Freescale Semiconductor Inc, 6501 William Cannon Drive West, Austin, Texas 78735, USA, http://www.freescale.com

[62] Motorola, DSP56309 semiconductor data sheet.

[63] Motorola, *DSP56309 User's Manual.*

[64] Motorola, *DSP56300 24-BIT DIGITAL SIGNAL PROCESSOR FAMILY MANUAL.*

[65] Motorola, *DSP56303EVM user's manual*, rev 2.

[66] Motorola, DSP56303 semiconductor data sheet.

[67] Crystal Semiconductor Corporation, CS4215 data sheet.

[68] Motorola, *Suite56 DSP Tools User's Manual*, Release 6.3.

[69] J. Hyde, *USB Design by Example*, Wiley Computer Publishing, 1999.

[70] J. Axelson, *USB COMPLETE,* Lakeview Research, 1999.

[71] http://www.usb.org

[72] Compaq, Intel, Microsoft, NEC, *Universal Serial Bus Specification*, Revision 1.1, 1998.

[73] Philips, PDIUSBD12 integrated circuits data sheet, 1998.

[74] Philips, *D12 SMART USER'S MANUAL.*

[75] Philips, *Firmware Programming Guide for PDIUSBD12*, Version 1.

[76] Altium, 12A Rodborough Rd, Frenchs Forest NSW 2086, Australia, http://www.altium.com

[77] IDT, 71V256SA data sheet.

[78] Atmel, AT29LV020 data sheet.

[79] PACE, Inc. 9030 Junction Drive, Annapolis Junction MD 20701, U.S.A, http://www.paceworldwide.com

[80] Microsoft Corporation, One Microsoft Way, Redmond, Washington 98052-6399, U.S.A, http://www.microsoft.com

[81] Motorola, *DSP563CCC MOTOROLA DSP56300 FAMILY OPTIMIZING C COMPILER USER'S MANUAL.*

[82] Motorola, *MOTOROLA DSP LINKER/LIBRARIAN REFERENCE MANUAL.*

[83] Motorola, *DSP ASSEMBLER REFERENCE MANUAL.*

[84] Microsoft, *WindowsXP 2600 Driver Development Kit.*

[85] W. Oney, *Programming the Microsoft Windows Driver Model*, 1999.

[86] Metrowerks, 7700 West Parmer Lane, Austin, Texas 78729, U.S.A http://www.metrowerks.com

[87] PROSPA, Data analysis software developed by C. Eccles, Massey University, 1999.

[88] Analog Devices, AD6644 data sheet.

[89] Analog Devices, AD8370 data sheet.

[90] Analog Devices, One Technology Way, Norwood, MA 02062-9106, U.S.A, http://www.analog.com

[91] Intersil Corporation Headquarters, 1001 Murphy Ranch Road, Milpitas, CA 95035, http://www.intersil.com

[92] Graychip, Inc, 2185 Park Blvd, Palo Alto, California 94306, USA, http://www.graychip.com

[93] Analog Devices, AD6620 data sheet.

[94] E. B. Hogennauer, IEEE Transactions on Acoustics, Speech, and Digital Signal Processing, Vol. ASSP-29, No. 2, April, 1981.

[95] R. G. Lyons, *Understanding Digital Signal Processing*, Prentice Hall, 2004.

[96] Analog Devices, *Designing filters for the AD6620.*

[97] Analog Devices, AD8131 data sheet.

[98] Analog Devices, AD9852 data sheet.

[99] Analog Devices, ADG901 data sheet.

[100] On Semiconductor, MC100EPT20 data sheet.

[101] On Semiconductor, MC100EP139 data sheet.

[102] On Semiconductor, MC100EP52 data sheet.

[103] On Semiconductor, MC100EPT23 data sheet.

[104] Fairchild Semiconductor, 74LCX74 data sheet.

[105] Fairchild Semiconductor, 74LCX08 data sheet.

[106] Fairchild Semiconductor, 74LCX32 data sheet.

[107] Analog Devices, Application note, AN501.

[108] Connor Winfield, CWX823 product data sheet.

[109] Mini-Circuits, Brooklyn, NY 11235, U.S.A, http://www.minicircuits.com

[110] Analog Devices AD8369 data sheet.

[111] Tomco Technologies, 17 Clarke Street, Norwood, SA 5067, Australia, http://www.tomco.com.au

[112] Miteq, Bipolar Amplifiers data sheet.

[113] Philips, BAV21 data sheet.

[114] E. Fukushima, S. B. W. Roeder, *Experimental Pulse NMR*, Westview Press, 1981.

[115] Schroff GmbH, Langenalber Straße 96-100, 75334 Straubenhardt, Germany, http://www.schroff.de

[116] TRACO Electronic AG, Jenatschstrasse 1, CH - 8002 Zurich, Switzerland, http://www.tracopower.com

[117] National Semiconductor, LM2599 data sheet.

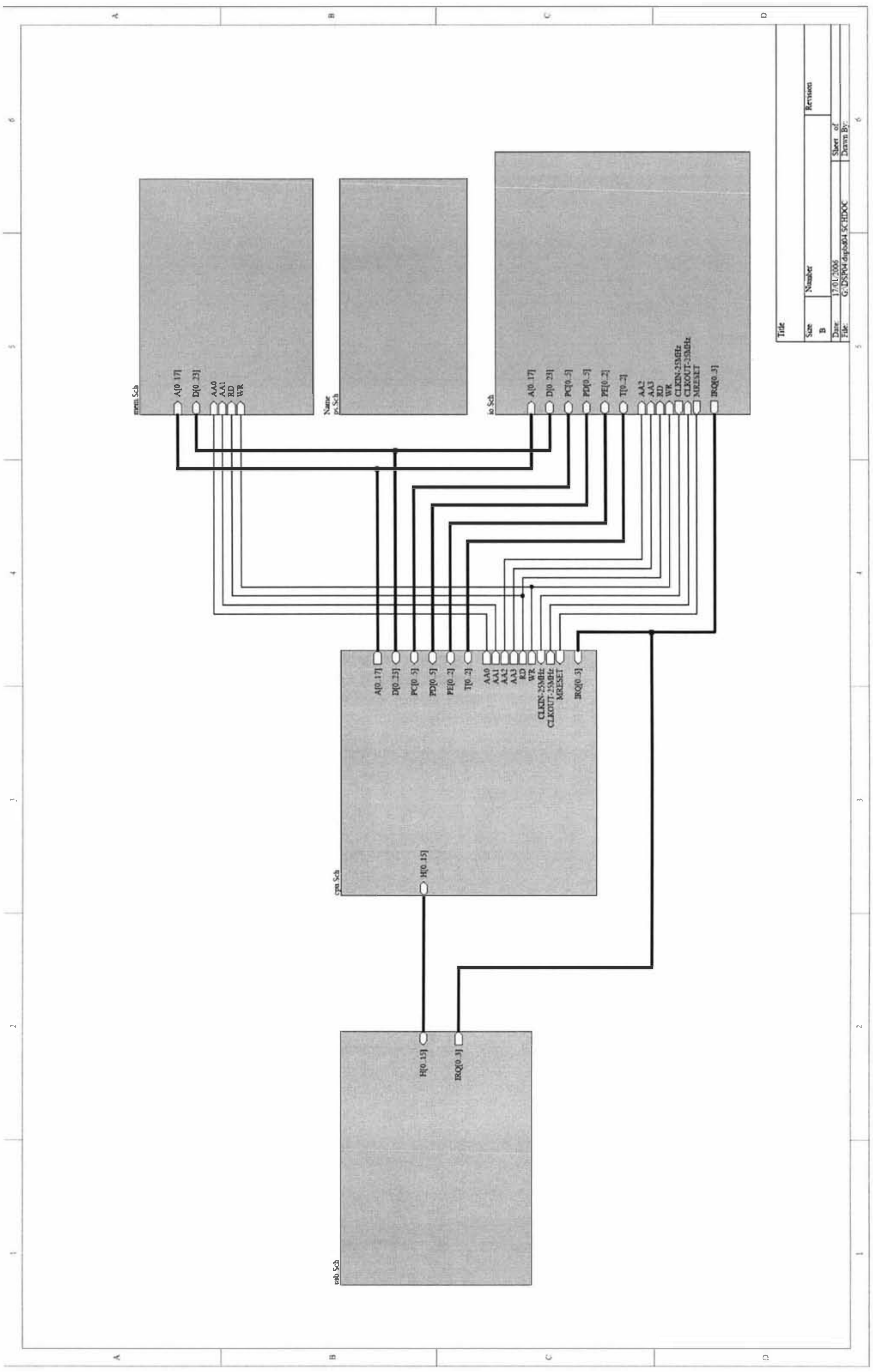
[118] Hammond Manufacturing Company Limited, 394 Edinburgh Rd North, Guelph , ON N1H 1E5, Canada, http://www.hammondmfg.com

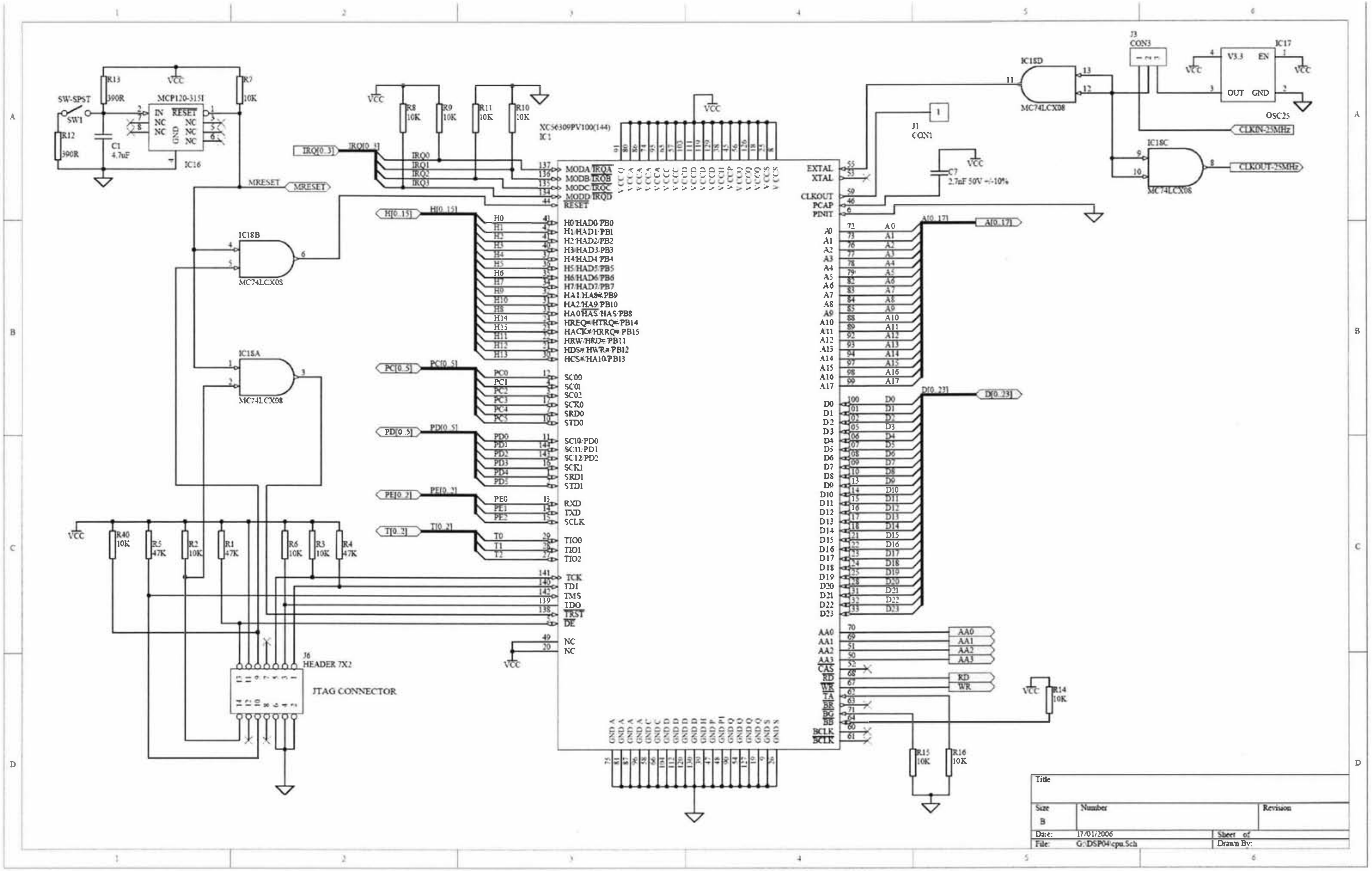
[119] 4th Colloquium on Mobile NMR, Aachen, Germany, 23-24 September, 2004.

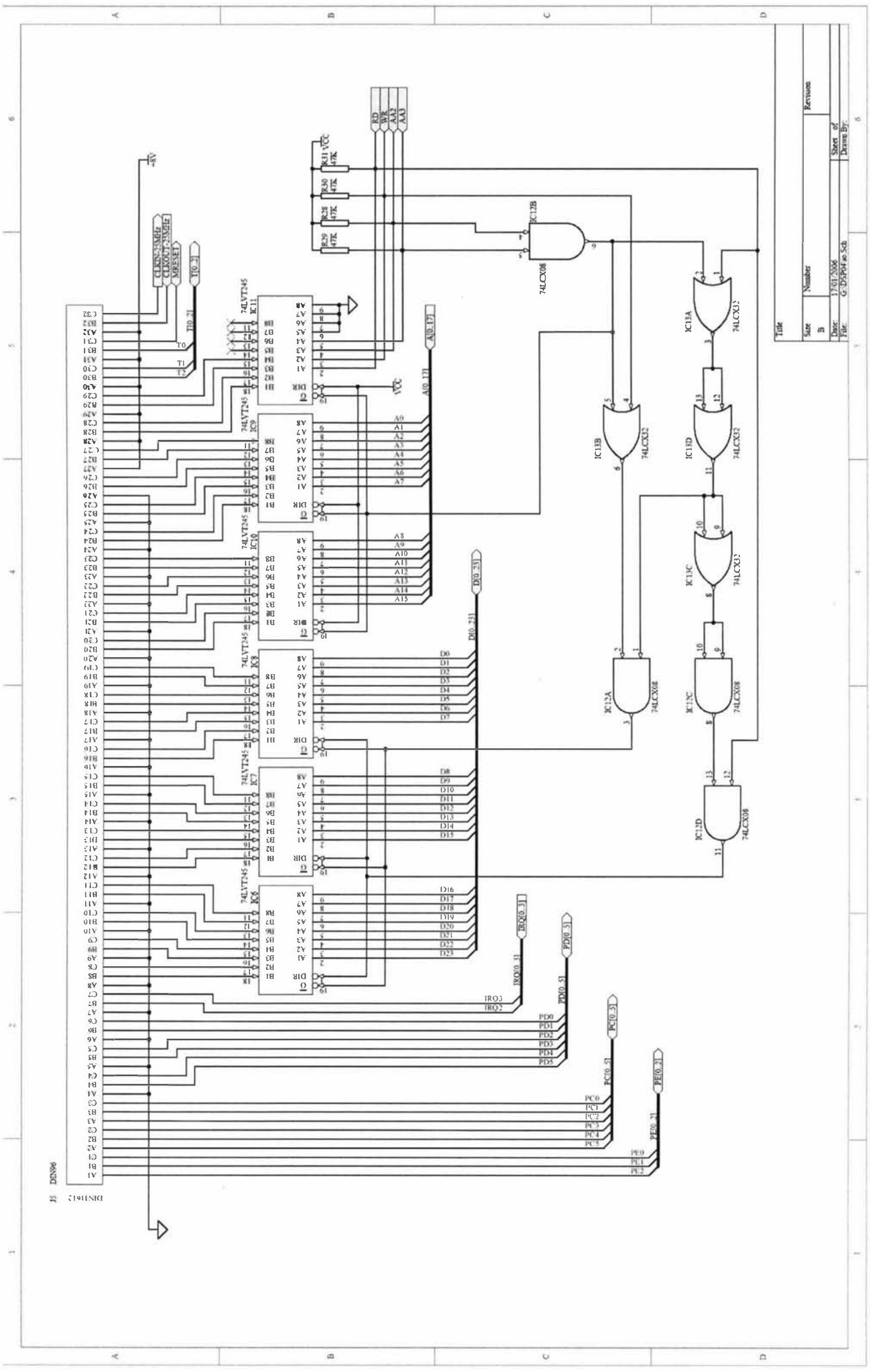
[120] 5th Colloquium on Mobile NMR, Perugia, Italy, 21-23 September, 2005.

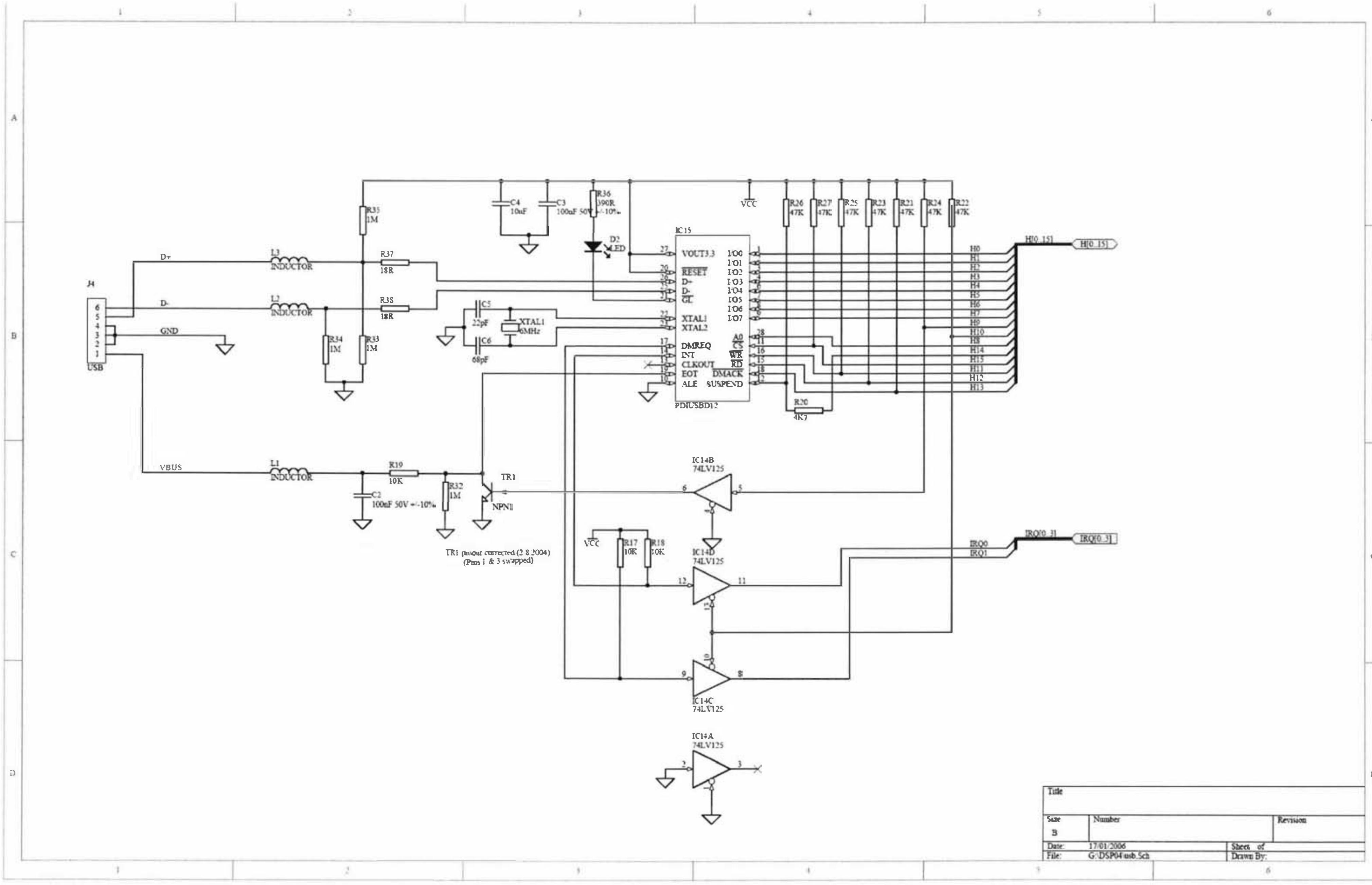
Appendix A1

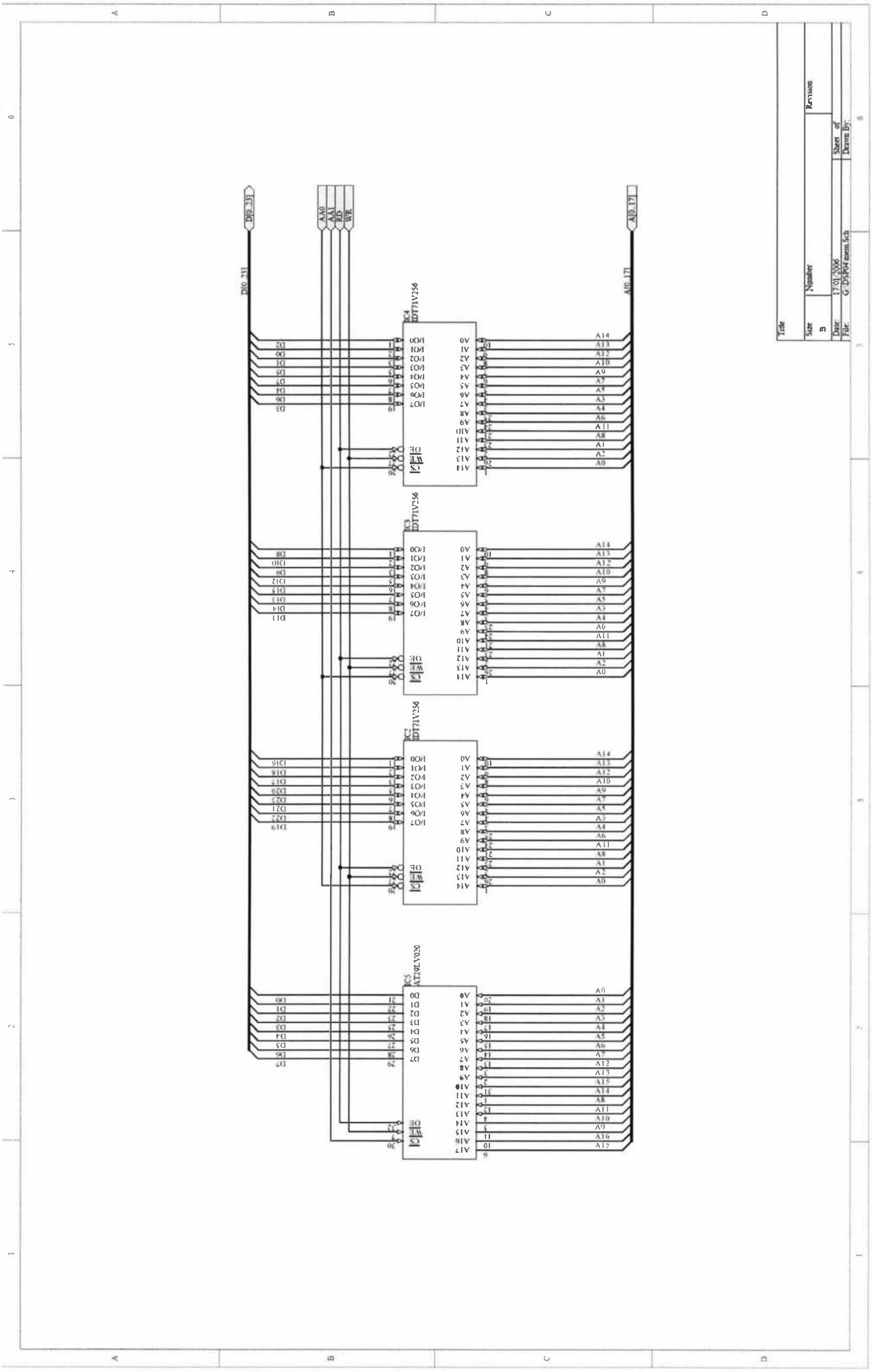
DSP/USB board schematics

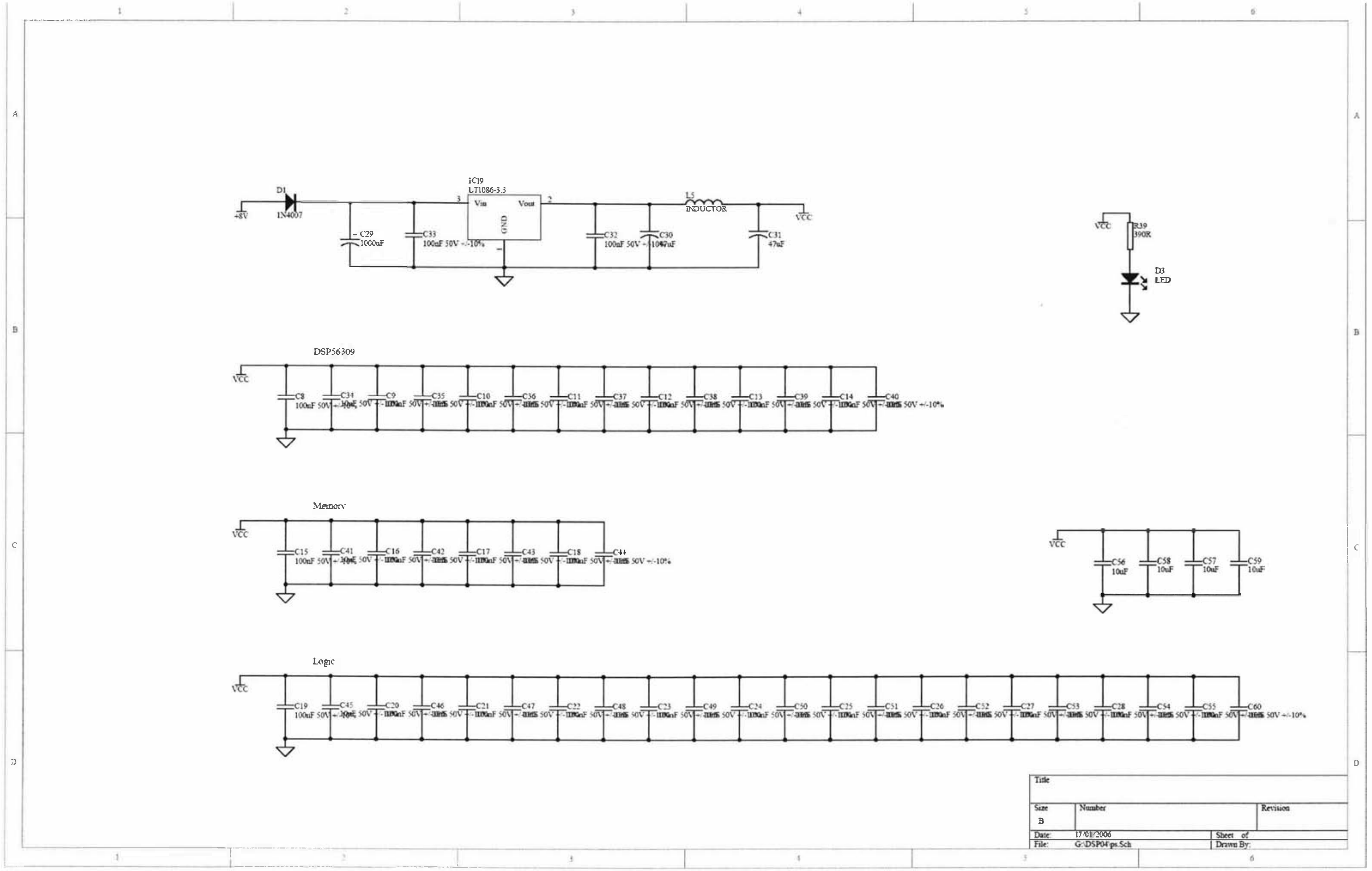

mem.Sch
A[0..17]
D[0..23]
AA0
AA1
RD
WR
Name
ps.Sch
io.Sch
A[0..17]
D[0..23]
PC[0..5]
PD[0..5]
PE[0..2]
T[0..2]
AA2
AA3
RD
WR
CLKIN-25MHz
CLKOUT-25MHz
MRESET
IRQ[0..3]
cpu.Sch
A[0..17]
D[0..23]
PC[0..5]
PD[0..5]
PE[0..2]
T[0..2]
AA0
AA1
AA2
AA3
RD
WR
CLKIN-25MHz
CLKOUT-25MHz
MRESET
IRQ[0..3]
H[0..15]
usb.Sch
H[0..15]
IRQ[0..3]
Title
Size
B
Number
Revision
Date:
17/01/2006
Sheet of
File:
G:\DSP04\dspb04.SCHDOC
Drawn By:


XC56309PV100(144)
IC1
MCP120-315I
IC16
SW-SPST
SW1
MC74LCX08
IC18A
IC18B
IC18C
IC18D
IC17
OSC25
CLKIN-25MHz
CLKOUT-25MHz
J3
CON3
J1
CON1
C7
2.7nF 50V +/-10%
J6
HEADER 7X2
JTAG CONNECTOR
MRESET
IRQ[0..3]
H[0..15]
PC[0..5]
PD[0..5]
PE[0..2]
TI[0..2]
A[0..17]
D[0..23]
AA0
AA1
AA2
AA3
RD
WR
VCC
Title
Size
B
Number
Revision
Date:
17/01/2006
Sheet of
File:
G:\DSP04\cpu.Sch
Drawn By:


Title
Size B
Number
Revision
Date: 17/01/2006
File: G:\DSP04\so.Sch
Sheet of
Drawn By:
J5 DIN96
DIN41612
CLKIN-25MHz
CLKOUT-25MHz
MRESET
T[0..2]
-5V
VCC
RD
WR
AA2
AA3
R31 47K
R30 47K
R28 47K
R29 47K
IC12B 74LCX08
IC13A 74LCX32
IC13B 74LCX32
IC13D 74LCX32
IC13C 74LCX32
IC12A 74LCX08
IC12C 74LCX08
IC12D 74LCX08
74LVT245 IC11
74LVT245 IC9
74LVT245 IC10
74LVT245 IC8
74LVT245 IC7
74LVT245 IC6
A[0..17]
D[0..23]
IRQ[0..3]
PD[0..5]
PC[0..5]
PE[0..2]

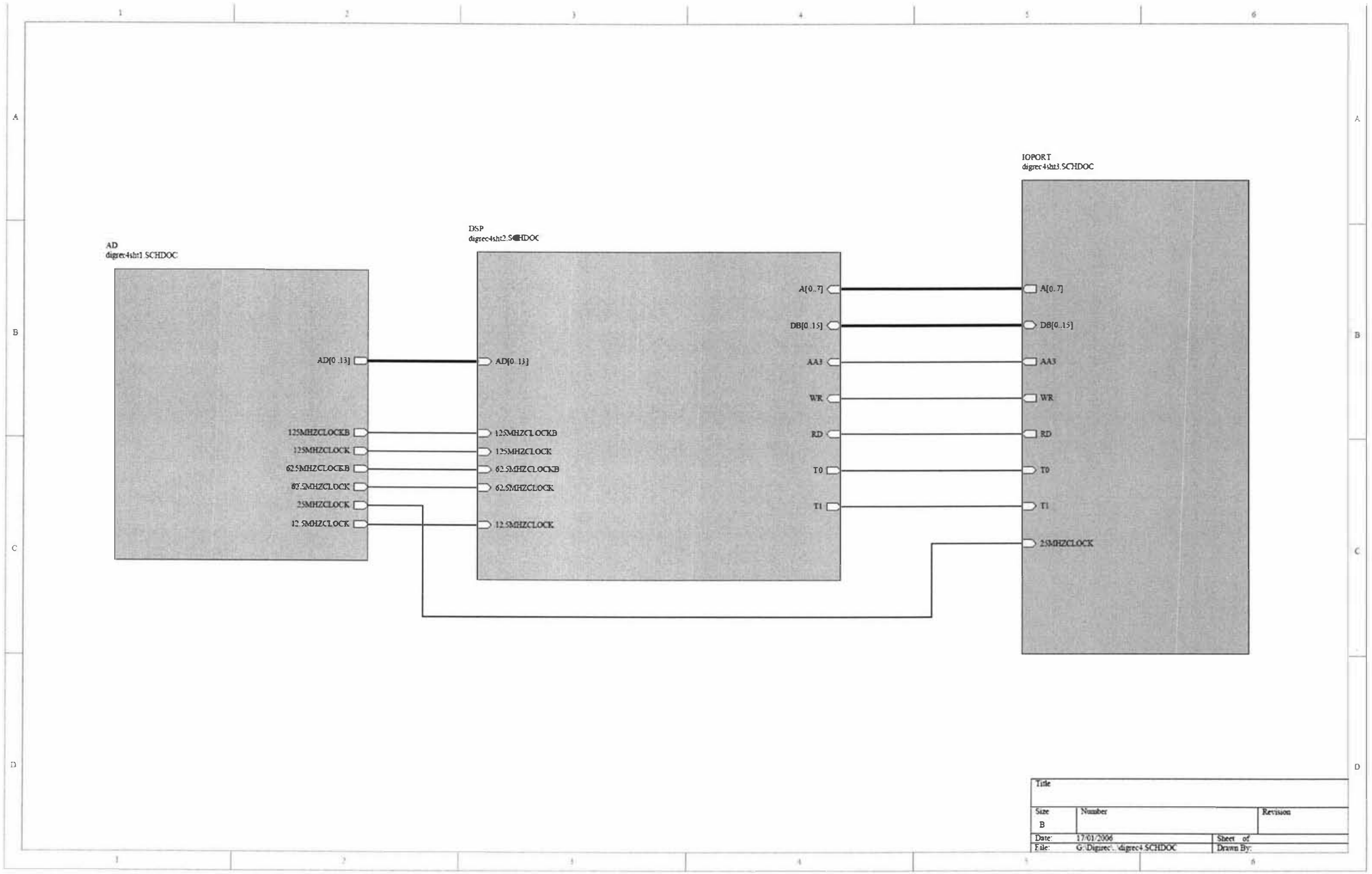

J4
USB
D+
D-
GND
VBUS
L1 INDUCTOR
L2 INDUCTOR
L3 INDUCTOR
R35 1M
R37 18R
R38 18R
R34 1M
R33 1M
R19 10K
R32 1M
C2 100nF 50V +/-10%
TR1
NPN1
TR1 pinout corrected (2.8.2004) (Pins 1 & 3 swapped)
C4 10uF
C3 100nF 50V +/-10%
R36 390R
D2 LED
C5 22pF
C6 68pF
XTAL1 6MHz
VCC
IC15
PDIUSBD12
VOUT3.3
RESET
GL
XTAL1
XTAL2
DMREQ
INT
CLKOUT
EOT
ALE
I/O0
I/O1
I/O2
I/O3
I/O4
I/O5
I/O6
I/O7
A0
CS
WR
RD
DMACK
SUSPEND
R26 47K
R27 47K
R25 47K
R23 47K
R21 47K
R24 47K
R22 47K
R20 4K7
H[0..15]
H0
H1
H2
H3
H4
H5
H6
H7
H9
H10
H8
H14
H15
H11
H12
H13
IC14B 74LV125
IC14D 74LV125
IC14C 74LV125
IC14A 74LV125
R17 10K
R18 10K
IRQ[0..3]
IRQ0
IRQ1
Title
Size
B
Number
Revision
Date: 17/01/2006
File: G:\DSP04\usb.Sch
Sheet of
Drawn By:

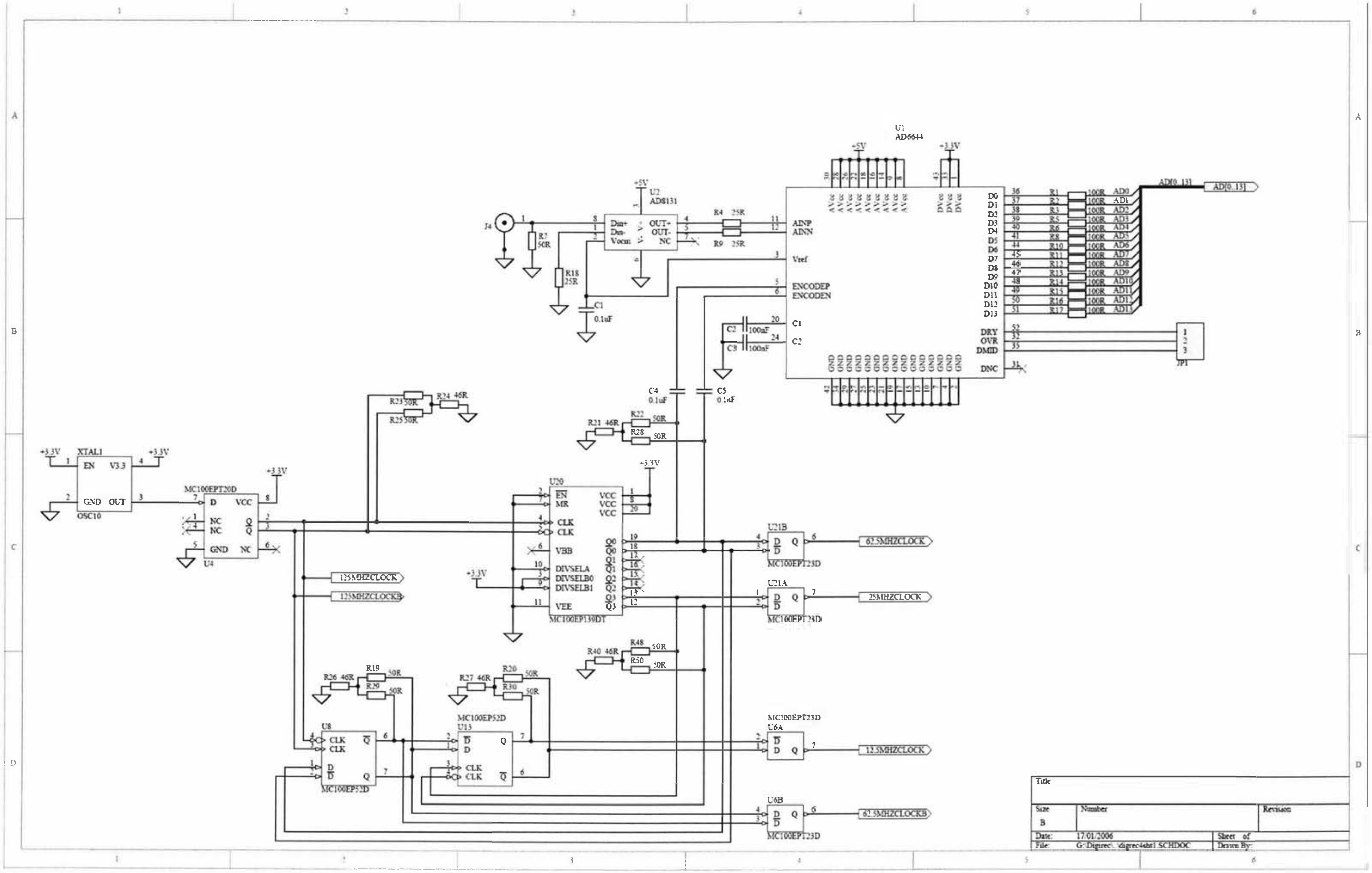

D[0..23]
AA0
AA1
RD
WR
A[0..17]
IC4 IDT71V256
IC3 IDT71V256
IC2 IDT71V256
IC5 AT29LV020
Title
Size B
Number
Revision
Date: 17/01/2006
File: G:\DSP04 mem.Sch
Sheet of
Drawn By:

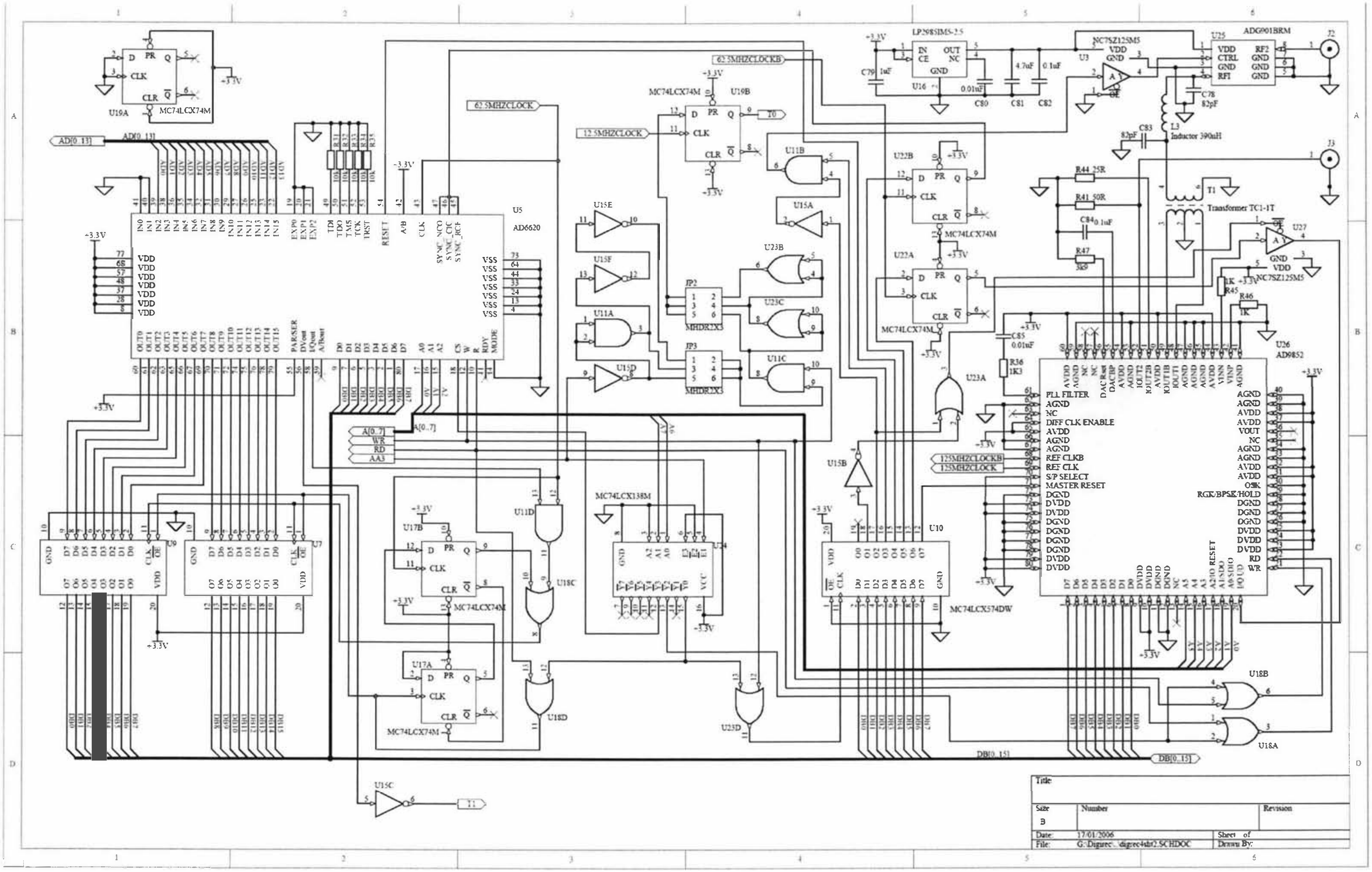

D1
1N4007
+8V
IC19
LT1086-3.3
Vin
Vout
GND
C29
1000uF
C33
100nF 50V +/-10%
C32
100nF 50V +/-10%
C30
47uF
L5
INDUCTOR
C31
47uF
VCC
R39
390R
D3
LED
DSP56309
Memory
Logic
C56
10uF
C58
10uF
C57
10uF
C59
10uF
Title
Size
B
Number
Revision
Date:
17/01/2006
Sheet of
File:
G:\DSP04\ps.Sch
Drawn By:

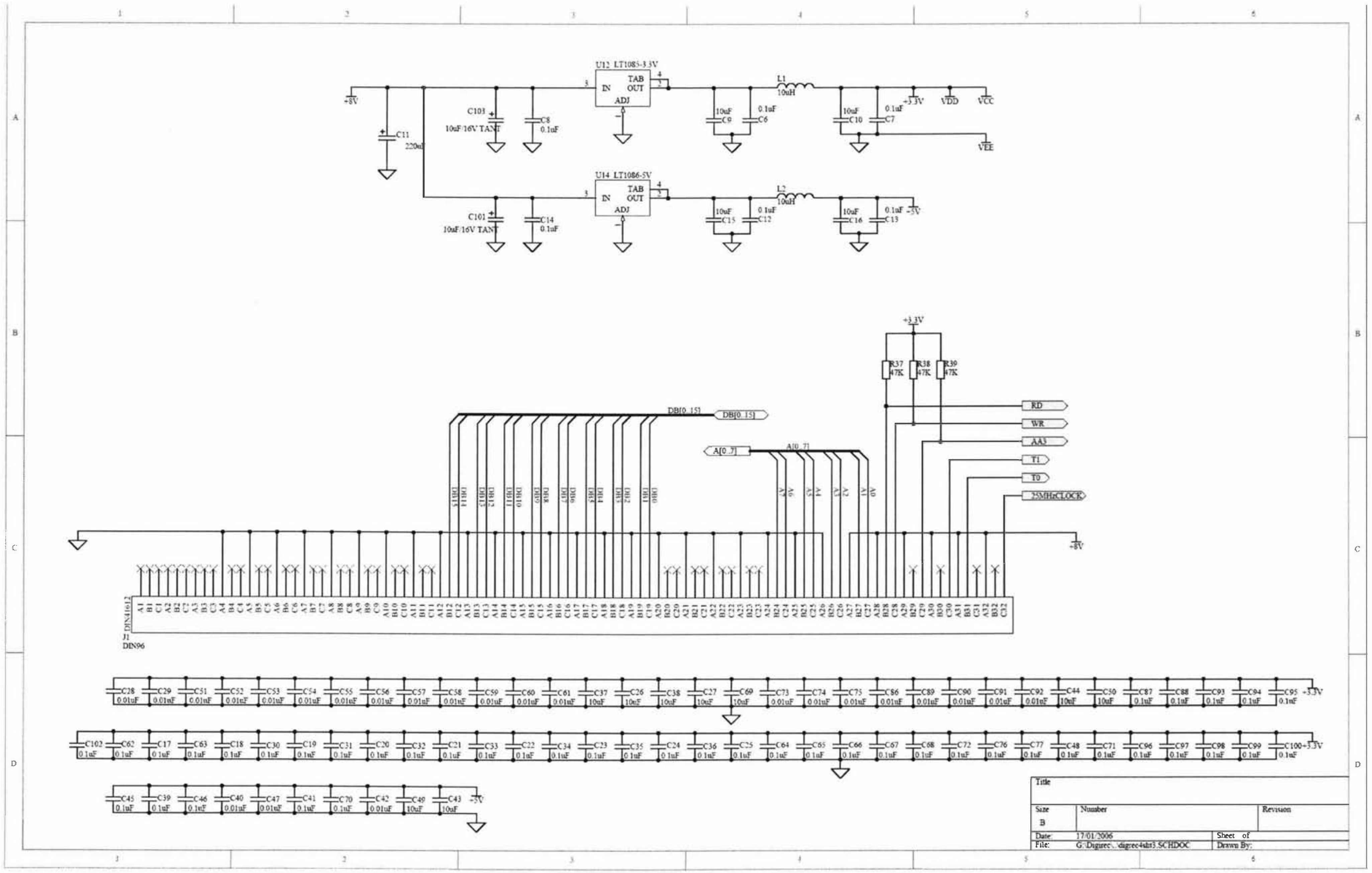
Appendix A2

Transceiver board schematics


1
2
3
4
5
6
A
B
C
D
AD
digrec4sht1.SCHDOC
AD[0..13]
125MHZCLOCKB
125MHZCLOCK
62.5MHZCLOCKB
62.5MHZCLOCK
25MHZCLOCK
12.5MHZCLOCK
DSP
digrec4sht2.SCHDOC
AD[0..13]
125MHZCLOCKB
125MHZCLOCK
62.5MHZCLOCKB
62.5MHZCLOCK
12.5MHZCLOCK
A[0..7]
DB[0..15]
AA3
WR
RD
T0
T1
IOPORT
digrec4sht3.SCHDOC
A[0..7]
DB[0..15]
AA3
WR
RD
T0
T1
25MHZCLOCK
Title
Size
B
Number
Revision
Date:
17/01/2006
Sheet of
File:
G:\Digirec\..\digrec4.SCHDOC
Drawn By:

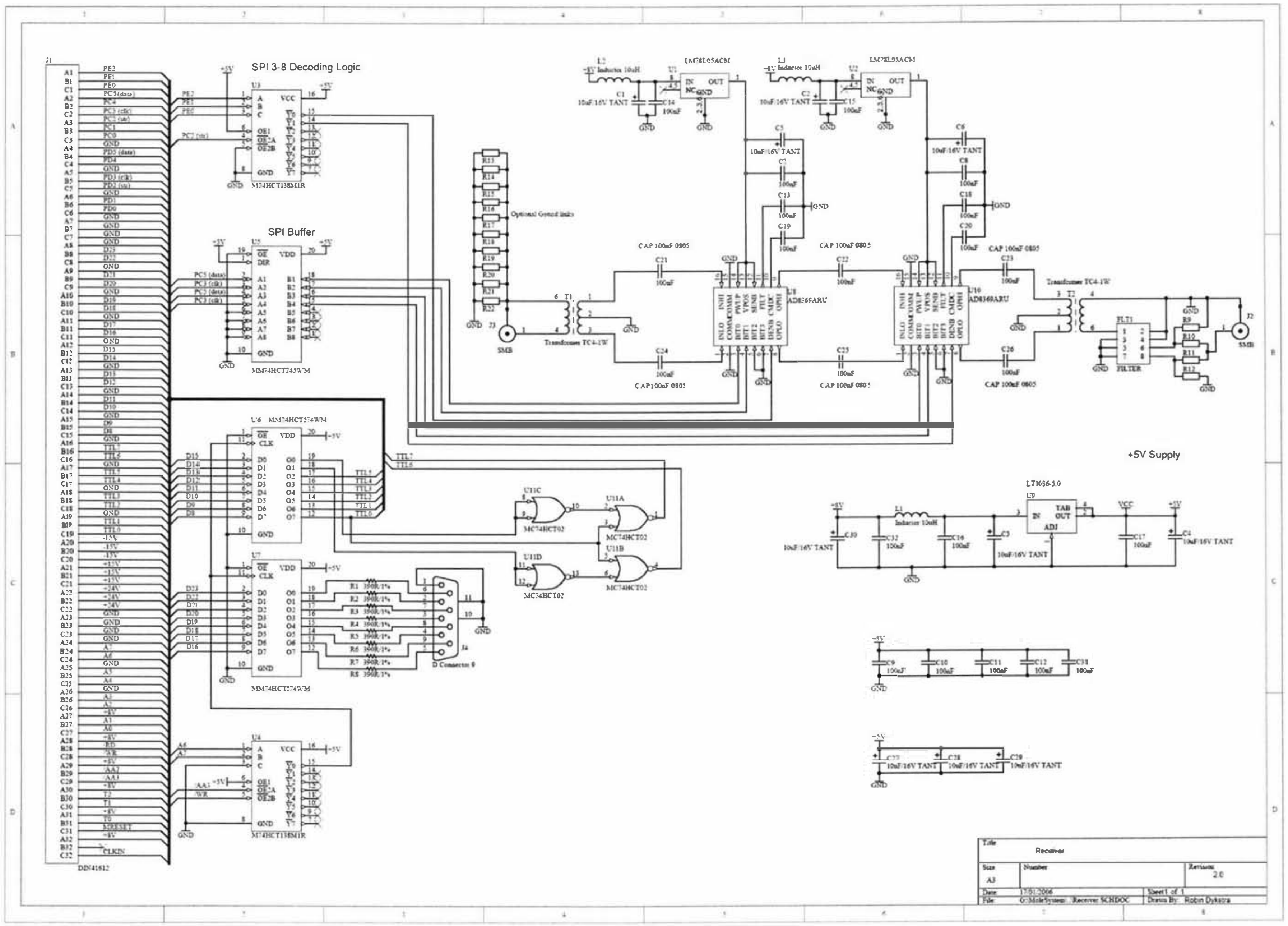

U1
AD6644
+5V
+3.3V
U2
AD8131
J4
R7 50R
R18 25R
C1 0.1uF
R4 25R
R9 25R
Din+
Din-
Vocm
OUT+
OUT-
NC
AINP
AINN
Vref
ENCODEP
ENCODEN
C1
C2
C2 100nF
C3 100nF
AVcc
DVcc
GND
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
DRY
OVR
DMID
DNC
R1
R2
R3
R5
R6
R8
R10
R11
R12
R13
R14
R15
R16
R17
100R
AD0
AD1
AD2
AD3
AD4
AD5
AD6
AD7
AD8
AD9
AD10
AD11
AD12
AD13
AD[0..13]
JP1
C4 0.1uF
C5 0.1uF
R21 46R
R22 50R
R28 50R
R23 50R
R25 50R
R24 46R
XTAL1
EN
V3.3
GND
OUT
OSC10
MC100EPT20D
D
VCC
NC
GND
U4
U20
EN
MR
CLK
VBB
DIVSELA
DIVSELB0
DIVSELB1
VEE
Q0
Q1
Q2
Q3
MC100EP139DT
125MHZCLOCK
125MHZCLOCKB
U21B
U21A
MC100EPT23D
62.5MHZCLOCK
25MHZCLOCK
R40 46R
R48 50R
R50 50R
R26 46R
R19 50R
R29 50R
R27 46R
R20 50R
R30 50R
U8
MC100EP52D
U13
Q
U6A
U6B
12.5MHZCLOCK
62.5MHZCLOCKB
Title
Size
B
Number
Revision
Date:
17/01/2006
Sheet of
File:
G:\Digirec\..\digirec4sht1.SCHDOC
Drawn By:


U19A MC74LCX74M
AD[0..13]
62.5MHZCLOCK
62.5MHZCLOCKB
12.5MHZCLOCK
U5 AD6620
U19B MC74LCX74M
T0
U11B
U15A
U15E
U15F
U11A
U15D
U23B
U23C
U11C
JP2 MHDR2X3
JP3 MHDR2X3
LP2985IM5-2.5
U16
C79 1uF
C80 0.01uF
C81 4.7uF
C82 0.1uF
NC7SZ125M5
U3
U25 ADG901BRM
J2
J3
C78 82pF
C83 82pF
L3 Inductor 390nH
T1 Transformer TC1-1T
R44 25R
R41 50R
C84 0.1uF
R47 3k9
U22B MC74LCX74M
U22A MC74LCX74M
U27 NC7SZ125M5
R45 1K
R46 1K
C85 0.01uF
R36 1K3
U26 AD9852
U23A
U15B
125MHZCLOCKB
125MHZCLOCK
A[0..7]
WR
RD
AA3
U9
U7
U11D
U17B MC74LCX74M
U18C
U17A MC74LCX74M
U18D
MC74LCX138M
U24
U10 MC74LCX574DW
U23D
U18B
U18A
DB[0..15]
U15C
T1
Title
Size B
Number
Revision
Date: 17/01/2006
Sheet of
File: G:\Digirec\..\digrec4sht2.SCHDOC
Drawn By:


U12 LT1085-3.3V
U14 LT1086-5V
IN
OUT
TAB
ADJ
+8V
C11 220uF
C103 10uF/16V TANT
C101 10uF/16V TANT
C8 0.1uF
C14 0.1uF
L1 10uH
L2 10uH
+3.3V
VDD
VCC
VEE
-5V
R37 47K
R38 47K
R39 47K
DB[0..15]
A[0..7]
RD
WR
AA3
T1
T0
25MHzCLOCK
J1 DIN96
DIN41612
Title
Size B
Number
Revision
Date: 17/01/2006
Sheet of
File: G:\Digirec\..\digrec4sht3.SCHDOC
Drawn By:

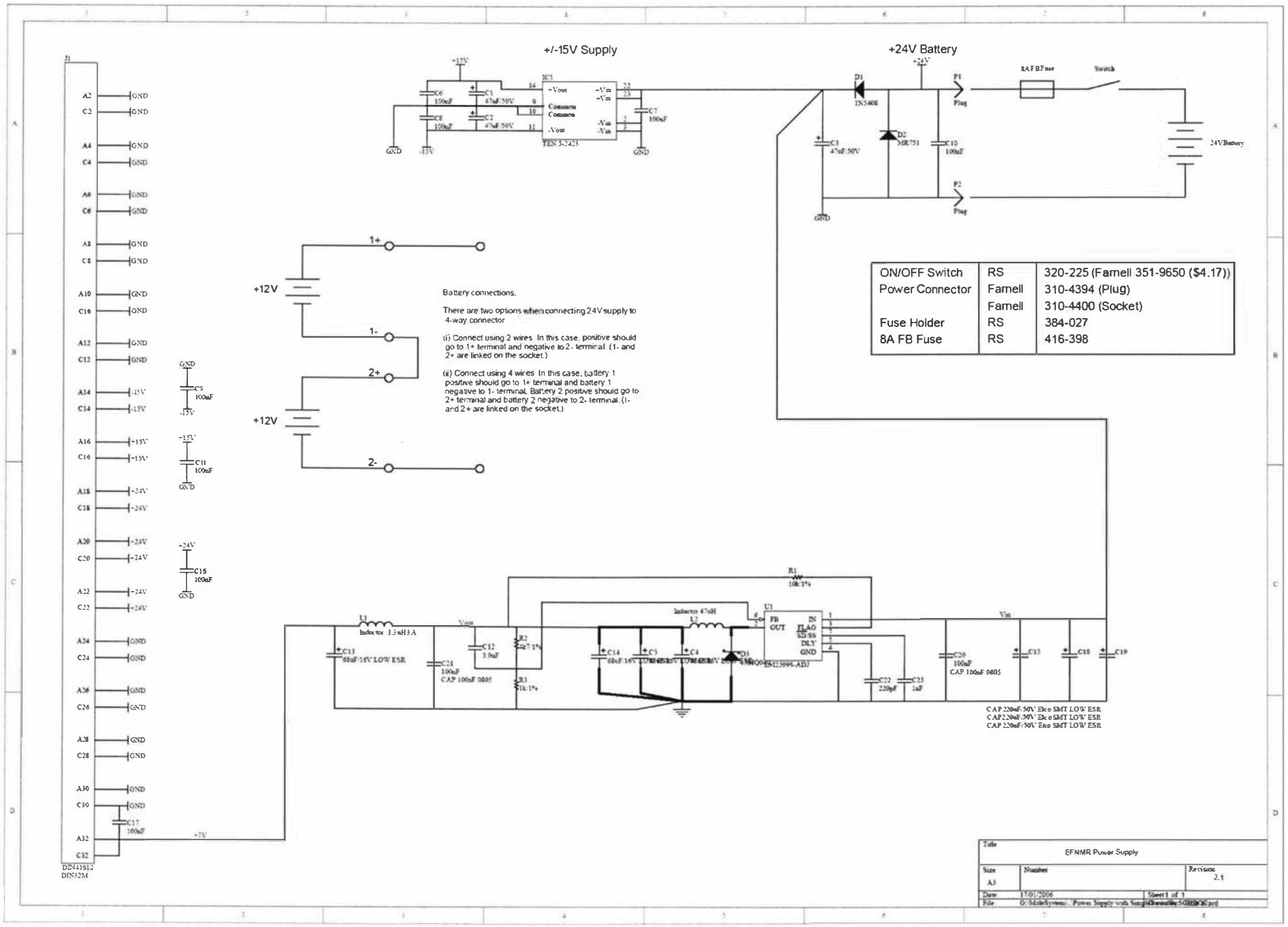
Appendix A3

Receiver amplifier board schematic


SPI 3-8 Decoding Logic
SPI Buffer
+5V Supply
M74HCT138M1R
MM74HCT245WM
MM74HCT574WM
MC74HCT02
AD8369ARU
LM78L05ACM
LT1086-3.0
Transformer TC4-1W
Optional Gound links
D Connector 9
DIN41612
SMB
FILTER
Title Receiver
Size A3
Revision 2.0
Date 17/01/2006
Sheet 1 of 1
File G:\MoleSystem\...\Receiver.SCHDOC
Drawn By: Robin Dykstra

Appendix A4

Modspec power supply board schematic



Battery connections.

There are two options when connecting 24V supply to 4-way connector

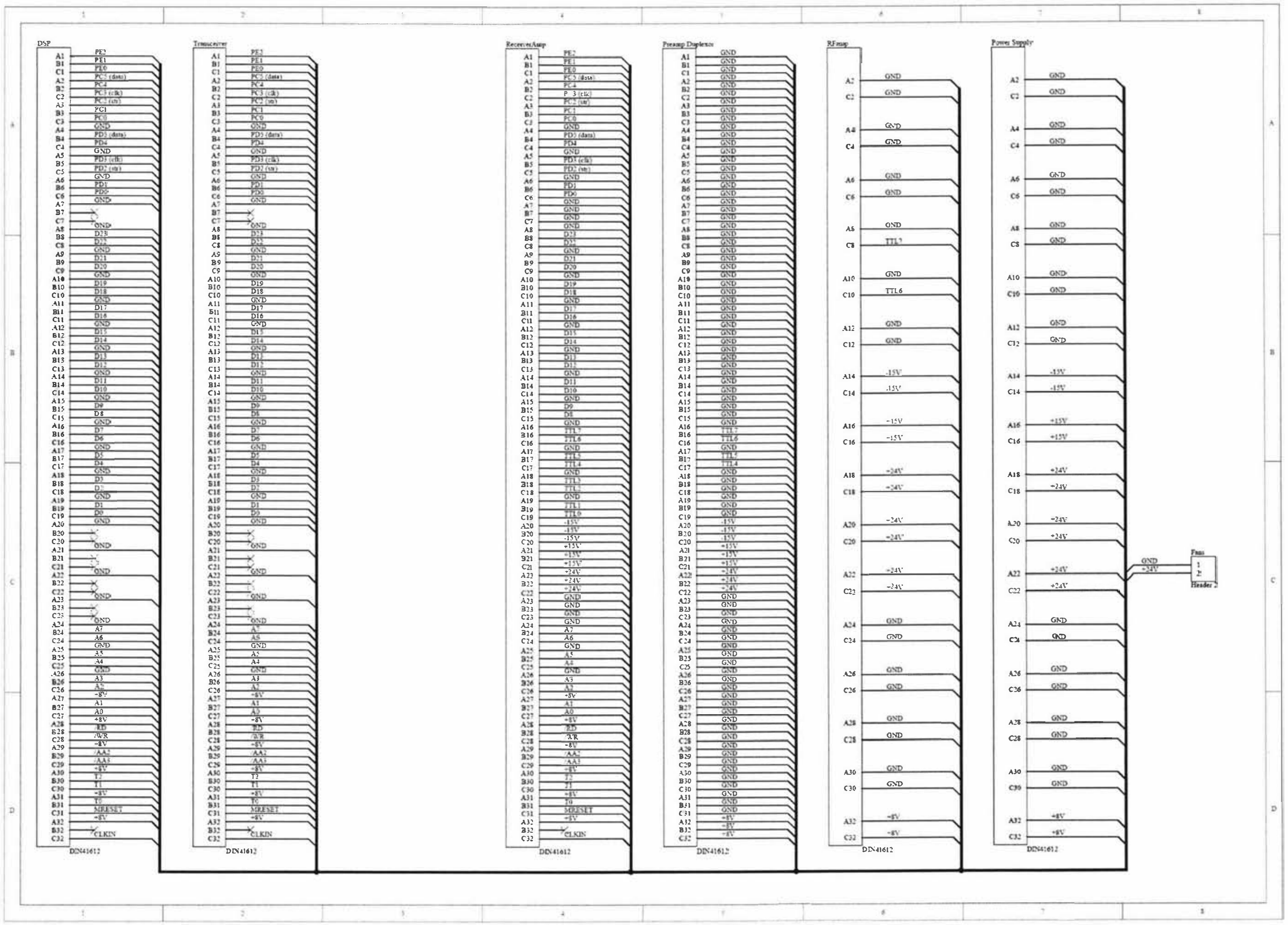
(i) Connect using 2 wires. In this case, positive should go to 1+ terminal and negative to 2- terminal. (1- and 2+ are linked on the socket.)

(ii) Connect using 4 wires. In this case, battery 1 positive should go to 1+ terminal and battery 1 negative to 1- terminal. Battery 2 positive should go to 2+ terminal and battery 2 negative to 2- terminal. (1- and 2+ are linked on the socket.)

ON/OFF Switch	RS	320-225 (Farnell 351-9650 ($4.17))
Power Connector	Farnell	310-4394 (Plug)
	Farnell	310-4400 (Socket)
Fuse Holder	RS	384-027
8A FB Fuse	RS	416-398

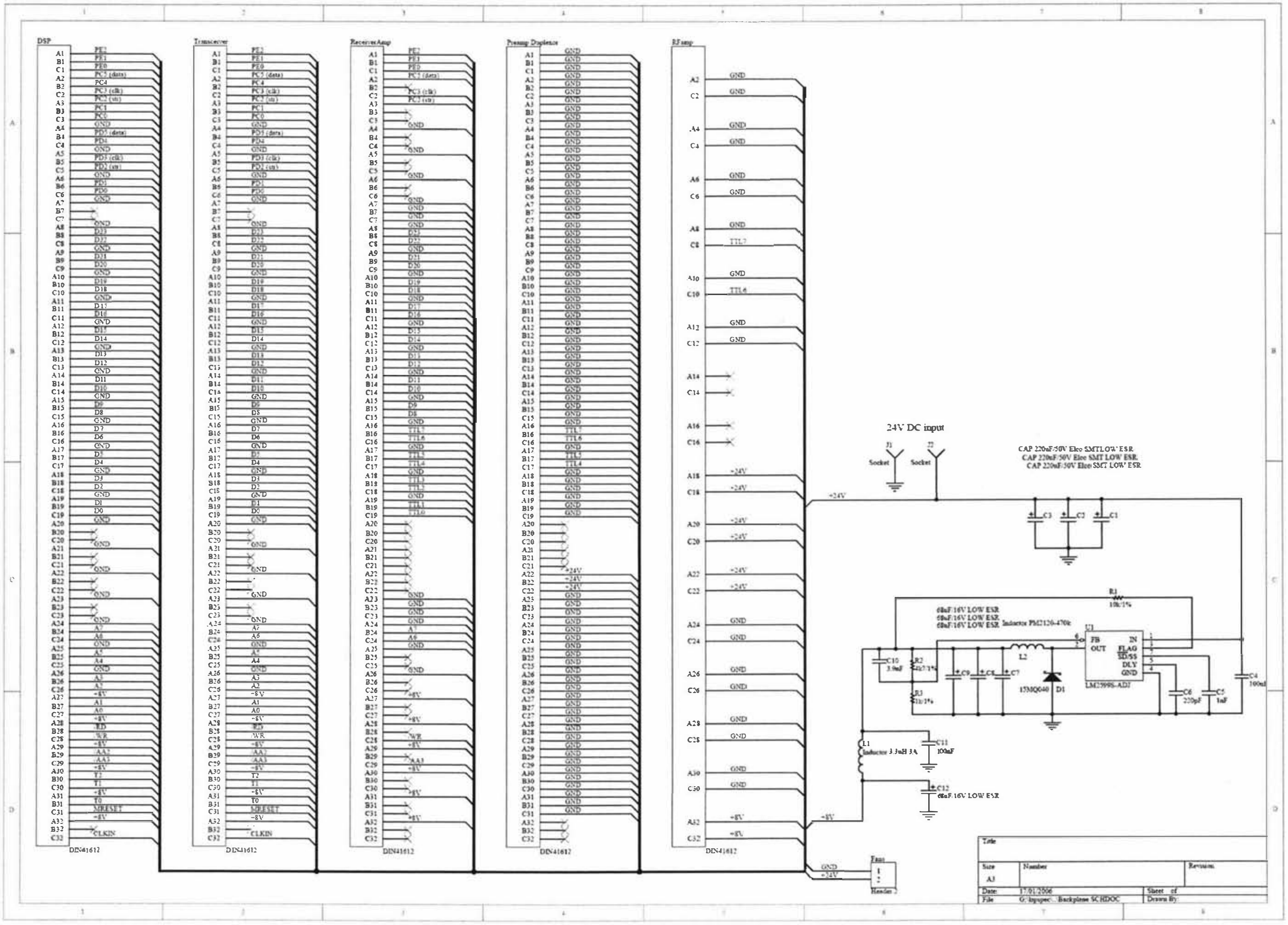
Appendix A5

Modspec backplane board schematic


DSP
Transceiver
ReceiverAmp
Preamp Duplexer
RFamp
Power Supply
DIN41612
Fans
Header 2

Appendix A6

Lapspec backplane board schematic


DSP
Transceiver
ReceiverAmp
Preamp Duplexer
RFamp
DIN41612
24V DC input
Socket
CAP 220uF/50V Elco SMT LOW ESR
68uF/16V LOW ESR
Inductor PM2120-470k
LM2599S-ADJ
15MQ040
Inductor 3.3uH 3A
Fans
Header 2
Title
Size
A3
Number
Revision
Date
17/01/2006
File
Sheet of
Drawn By

Appendix B1

DSP spin-echo pulse program

```
;************************************************************
;Echo
;For the 56309 DSP/USB board and digital transceiver board.
;Sample rate = 1/512ns
;
;Robin Dykstra
;13/5/05
;
;*********************************************************************
        nolist
        include  'ioequ.asm'
        list
;*********************************************************************

        org    x:$0

PARAM_BASE      equ     *       ; Memory for pulse program parameters
TXW0            ds      1       ; Tx word 0
TXW1            ds      1       ; Tx word 1
TXW2            ds      1       ; Tx word 2
TXW3            ds      1       ; Tx word 3
TXW4            ds      1       ; Tx word 4
RXFW0           ds      1       ; Rx Frequency word 0
RXFW1           ds      1       ; Rx Frequency word 1
RXFW2           ds      1       ; Rx Frequency word 2
RXFW3           ds      1       ; Rx Frequency word 3
RXPW0           ds      1       ; Rx Phase word 0
RXPW1           ds      1       ; Rx Phase word 1
Ndatapoints     ds      1       ; Number of data points to acquire
T90             ds      1       ; 90 Pulse time
T180            ds      1       ; 180 Pulse time
Techo           ds      1       ; Echo time
Grec            ds      1       ; Receiver gain
Tcycle          ds      1       ; Recycle delay

;*********************************************************************
;
;       Space for filter coeffs, samples etc
;
        org     y:$0

COEFF_BASE      equ     *       ;Filter coefficients

;*********************************************************************
;
        org     p:$001000           ;start

        move    r0,x:(r6)+          ;Save registers
        move    r1,x:(r6)+
        move    r2,x:(r6)+
        move    r3,x:(r6)+
        move    r4,x:(r6)+
        move    r5,x:(r6)+
        move    r7,x:(r6)+
        move    x0,x:(r6)+
        move    b0,x:(r6)+
        move    b1,x:(r6)+
        move    b2,x:(r6)+
        movep   #$0F3FE1,x:A_BCR    ;Set up wait states, 1 for AA3
        movep   #0,x:A_TLR0             ;Set up synch timer
        movep   #0,x:A_TCSR0            ;Disable timer
        movep   #$200261,x:A_TCSR0      ;Set up timer0 input capture
        move    #$80,a1
synchl  move    a1,x:$20000         ;Wait for synch
        jclr    #21,x:A_TCSR0,synchl
        nop
        movep   #$0FFFE1,x:A_BCR    ;Set up wait states, 7 for AA3
        nop

; Setup AD9852
        move    #$80,a1                   ;Reset 9852
        move    a1,x:$20000
        bsr     swait
        move    #$00,a1
        move    a1,x:$20000
        move    #$14,a1             ;Set up control regs
        move    a1,x:$2009D
        move    #$24,a1
        move    a1,x:$2009E
        move    #$00,a1
        move    a1,x:$2009F
        move    #$60,a1
        move    a1,x:$200A0
        move    x:TXW1,a1               ;Output W1 to 9852
        move    a1,x:$20084
        move    x:TXW2,a1               ;Output W2 to 9852
        move    a1,x:$20085
        move    x:TXW3,a1               ;Output W3 to 9852
        move    a1,x:$20086
        move    x:TXW4,a1               ;Output W4 to 9852
        move    a1,x:$20087
        move    #$0F,a1             ;output amplitude
        move    a1,x:$200A1
        move    #$FF,a1
        move    a1,x:$200A2
;
; Setup AD6620
;
        move    #$00,a1                 ;Reset 6620
        move    a1,x:$20000
        move    #$10,a1
        move    a1,x:$20000
        move    #$03,a1                 ;Put into soft reset
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        move    #$01,a1
        move    a1,x:$200C0
        move    #drecf,r1              ;Set up filter coeffs
        move    #$80,a1
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        do      #12,fcl             ;12 coeffs
        move    p:(r1),a1
        lsr     #16,a
        move    a1,x:$200C2
        move    p:(r1),a1
        lsr     #8,a
        move    a1,x:$200C1
        move    p:(r1)+,a1
        move    a1,x:$200C0
fcl
        move    #$81,a1                 ;Clear filter data
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        do      #256,fdl
        move    #$00,a1
        move    a1,x:$200C4
        move    a1,x:$200C3
        move    a1,x:$200C2
        move    a1,x:$200C1
        move    a1,x:$200C0
fdl
        move    #$83,a1                 ;Set up 6620 config regs
        move    a1,x:$200C7
        move    #$01,a1
        move    a1,x:$200C6
        move    #$00,a1                 ;NCO control reg
        move    a1,x:$200C0
        move    #$ff,a1                  ;NCO SYNC control reg
        move    a1,x:$200C3
        move    #$ff,a1
        move    a1,x:$200C2
        move    #$ff,a1
        move    a1,x:$200C1
        move    #$ff,a1
        move    a1,x:$200C0
        move    x:RXFW0,x0               ;NCO FREQ control reg
        move    x0,x:$200C3
        move    x:RXFW1,x0
        move    x0,x:$200C2
        move    x:RXFW2,x0
        move    x0,x:$200C1
        move    x:RXFW3,x0
        move    x0,x:$200C0
        move    #$00,a1                  ;NCO PHASE control reg
        move    a1,x:$200C1
        move    #$00,a1
        move    a1,x:$200C0
        move    #$02,a1                  ;INPUT/CIC2 scale reg
        move    a1,x:$200C0
        move    #$03,a1                  ;Mcic2-1 reg
        move    a1,x:$200C0
        move    #$0A,a1                  ;CIC5 scale reg
        move    a1,x:$200C0
        move    #$07,a1                  ;Mcic5-1 reg
        move    a1,x:$200C0
        move    #$04,a1                  ;OUTPUT/RCF control reg
        move    a1,x:$200C0
        move    #$00,a1                  ;Mrcf-1 reg
        move    a1,x:$200C0
        move    #$00,a1                  ;RCF ADDRESS OFFSET reg
        move    a1,x:$200C0
        move    #11,a1                   ;Ntaps-1 reg
        move    a1,x:$200C0
        move    #$00,a1                  ;Reserved reg
        move    a1,x:$200C0
        move    #$03,a1                  ;MODE CONTROL reg
        move    a1,x:$200C7
        move    #$00,a1
        move    a1,x:$200C6
        move    #$00,a1
        move    a1,x:$200C0

        nop
        movep   #$0F9FE1,x:A_BCR      ;Set up wait states, 4 for AA3
        nop

        move    #$12,a1               ;Turn on 9852 synch buffer
        move    a1,x:$20000
        bsr     swait
        move    #$72,a1                  ;Toggle SYNCs
        move    a1,x:$20000
        bsr     swait
        move    #$12,a1
        move    a1,x:$20000

; Setup Receiver gain
        move    #$2C,x:A_PCRC       ;Set up SSI 0
        movep   #$10080A,x:A_CRA0
        movep   #$13C3C,x:A_CRB0
;       move    x:Grec,a1
;       lsl     #12,a
        move    #$1111,a1           ;receiver gain 0000-FFFF
        movep   #$0000,x:A_PDRE     ;select first gain block
        movep   a,x:A_TX00          ;Set up first gain block
        move    #10,r7
        bsr     wait
        movep   #$0004,x:A_PDRE     ;select next gain block
        movep   a,x:A_TX00          ;Set up second gain block
        move    #10,r7
        bsr     wait
        movep   #$0007,x:A_PDRE           ;select unused
        movep   #$3c3c,x:A_CRB0           ;Shut down SSI 0
        movep   #$00,x:A_PCRC

;Output 90/180

        move    #$00,r1                 ;use r1 as the event counter
        movep   #0,x:A_TLR2             ;Set up event timer
        move    #100,r3
        movep   r3,x:A_TCPR2            ;Set for first event
        movep   #$A01,x:A_TCSR2


evenl   jclr    #21,x:A_TCSR2,evenl     ;Wait for event timer flag
        move    (r1)+                   ;Increment event counter
        move    r1,b
        cmp     #1,b
        beq     event1
        cmp     #2,b
        beq     event2
        cmp     #3,b
        beq     event3
        cmp     #4,b
        beq     event4
        cmp     #5,b
        beq     event5
        bra     event6

;Sample data

adcs    movep   #0,x:A_TLR1             ;Set up ADC timer
        movep   #0,x:A_TCSR1            ;Disable timer
        movep   #$361,x:A_TCSR1         ;Set up timer1 input capture
        move    #$10000,r5         ;Make r5 point to the start of fid memory
        move    #$0400,r7               ;Load the number of samples into r7
adcl    jclr    #21,x:A_TCSR1,adcl      ;Wait for timer1 flag
        movep   #$200361,x:A_TCSR1      ;Clear timer1 flag
        move    x:$20000,a1             ;Load data from channel A
        lsl     #8,a                    ;make it 24bit
        move    a1,y:(r5)+              ;Write to memory
        move    x:$20000,a1             ;Load data from channel B
        lsl     #8,a                    ;make it 24bit
        move    a1,y:(r5)+              ;Write to memory
        move    (r7)-                   ;Decrement count
        move    r7,b
        tst     b
        beq     simp_e                  ;Stop if =0
        bra     adcl                    ;Otherwise Loop again
simp_e

        move    #$80,a1           ;Reset 6620 & 9852 and turn off synch buff
        move    a1,x:$20000
        move    x:Tcycle,r7
        bsr     wait

        move    x:-(r6),b2              ;Restore registers
        move    x:-(r6),b1
        move    x:-(r6),b0
        move    x:-(r6),x0
        move    x:-(r6),r7
        move    x:-(r6),r5
        move    x:-(r6),r4
        move    x:-(r6),r3
        move    x:-(r6),r2
        move    x:-(r6),r1
        move    x:-(r6),r0
        rts
;
;****************************************************************
;       Event 1, Unblank RF amp
;
event1  move    #50,r3              ;1us delay
        movep   r3,x:A_TCPR2            ;Set for next event
        movep   #$200A01,x:A_TCSR2      ;Clear timer flag
        move    #$1C012,a1          ;Unblank RF amp
        move    a1,x:$20000
        bra     evenl
;
;****************************************************************
;       Event 2, Start TX 90 pulse
;
event2  move    x:T90,r3
        movep   r3,x:A_TCPR2            ;Set for next event
        movep   #$200A01,x:A_TCSR2      ;Clear timer flag
        move    #$1C01A,a1          ;start 90x Pulse
        move    a1,x:$20000
        bra     evenl
;
;****************************************************************
;       Event 3, Stop TX 90 Pulse and blank RF amp
;
event3  move    x:Techo,r3
        movep   r3,x:A_TCPR2            ;Set for next event
        movep   #$200A01,x:A_TCSR2      ;Clear timer flag
        move    #$1A,a1                 ;Stop 90 Pulse and blank RF amp
        move    a1,x:$20000
        move    #$20,a1             ;set phase of 9852 to y
        move    a1,x:$20080
        move    #$00,a1
        move    a1,x:$20081
        move    #$52,a1             ;Update phase by toggling 9852 synch
        move    a1,x:$20000
        move    #$12,a1
        move    a1,x:$20000
        bra     evenl
;
;****************************************************************
;       Event 4, Unblank RF amp
;
event4  move    #50,r3              ;1us delay
        movep   r3,x:A_TCPR2            ;Set for next event
        movep   #$200A01,x:A_TCSR2      ;Clear timer flag
        move    #$1C01A,a1          ;Unblank RF amp
        move    a1,x:$20000
        bra     evenl
;
;****************************************************************
;       Event 5, start TX 180 pulse
;
event5  move    x:T180,r3
```

```
        movep   r3,x:A_TCPR2             ;Set for next event
        movep   #$200A01,x:A_TCSR2       ;Clear timer flag
        move    #$1C01A,a1             ;start 180y Pulse
        move    a1,x:$20000
        bra     evenl
;
;************************************************************
;       Event 6, Stop TX 180 Pulse and blank RF amp
;
event6  movep   #$200A00,x:A_TCSR2       ;Clear timer flag and stop
        move    #$1A,a1                    ;Stop 180y Pulse and blank RF amp
        move    a1,x:$20000
                                       ;Reset 9852 to stop DDS?
        bra     adcs
;
;************************************************************
;       Wait n x 1us routine (100MHz clock)
;       On entry, r7 contains "n"
;
wait    rep     #90
        nop
        move    (r7)-
        move    r7,b
        tst     b
        bne     wait
        rts
;
;************************************************************
;       Wait 100ns routine (100MHz clock)
;
swait   rep     #4
        nop
        rts
;
;************************************************************
;
;       AD6620 parameters
;
drecp   dc      $00
;
;       AD6620 filter coefficients
;
drecf   dc    $03CDF
        dc    $F05BA
        dc    $ED13D
        dc    $FF4B3
        dc    $37B01
        dc    $67C66
        dc    $67C66
        dc    $37B01
        dc    $FF4B3
        dc    $ED13D
        dc    $F05BA
        dc    $03CDF
;
;************************************************************
```

Appendix B2

Prospa spin-echo experiment macro

```
###########################################################
#
# Spin Echo control macro
#
# Set up 62.5MHz digital receiver for 2MHz data output rate
# and 500kHz LPF cutoff.
###########################################################

procedure(echo)

# Initilaize value for parameters

   acqTime       = 131.072
   b1freq        = 20
   spare         = 0
   nrScans       = 1
   time90        = 2.6
   time180       = 5.2
   echoTime      = 40
   recycDelay    = 700
   recgain       = 50

   accum         = "yes"

   n = window("Echo experiment", 37, 485, 464, 201)
      windowvar(timeLeft,timeRight)
      timeLeft     = 0
      timeRight    = acqTime
      edittext(3, 327, 55, 42)
      statictext(4, 228, 58, "left", "Acquisition time (us)")
      edittext(7, 132, 55, 61)
      statictext(8, 27, 60, "left", "Acquisition delay (us)")
      edittext(9, 327, 78, 42)
      statictext(10, 241, 81, "left", "Number of scans")
      button(11, 392, 14, 52, 33, "RUN",
             enablecontrol(0,11,"false");
             enablecontrol(0,13,"false");
             echo:runpp();
             enablecontrol(0,11,"true");
             enablecontrol(0,15,"true");
             enablecontrol(0,13,"true");)
      button(13, 392, 98, 52, 33, "EXIT",
             closewindow(0);)
      groupbox(14, "Spin-echo parameters", 5, 8 , 197, 154)
      button(15, 392, 56, 52, 33, "STOP",
             enablecontrol(0,11,"true");
             enablecontrol(0,13,"true");
             enablecontrol(0,15,"true");)
      checkbox(25, 327, 106, "no,yes", "yes")
      statictext(26, 271, 107, "left", "Accumlate")
      edittext(27, 132, 78, 61)
      statictext(28, 20, 83, "left", "90 deg. pulse time (us)")
```

```
        edittext(1, 327, 32, 42)
        statictext(6, 245, 35, "left", "Receiver gain %")
        groupbox(33, "Acquisition parameters", 213, 8, 166, 154)
        edittext(34, 132, 32, 61)
        statictext(35, 32, 37, "left", "B1 frequency (MHz)")
        edittext(5, 132, 101, 61)
        statictext(12, 14, 106, "left", "180 deg. pulse time (us)")
        edittext(16, 132, 124, 61)
        statictext(18, 59, 129, "left", "Echo time (us)")
        edittext(17, 327, 124, 42)
        statictext(2, 232, 129, "left", "Recycle delay (ms)")
        setpar(0,15,"mode","panic");

   showobjects(n)

# Initialise pulse program parameters

   setpar(0,3,"text",acqTime)
   setpar(0,1,"text",recgain)
   setpar(0,7,"text",spare)
   setpar(0,9,"text",nrScans)
   setpar(0,16,"text",echoTime)
   setpar(0,17,"text",recycDelay)
   setpar(0,34,"text",b1freq)
   setpar(0,27,"text",time90)
   setpar(0,5,"text",time180)

endproc()

##############################################################
#
# Run the sequence
#
##############################################################

procedure(runpp)

  # Load pulse program and send to DSP

   cd("$appdir$\PulseSequences\Portable NMR\echo")
   pp = dspsrec("echo.p")
   dspwrite("p",0x1000,pp)

  # Extract pp parameters from GUI

   recgain      = getpar(0,1,"value")
   acqTime      = getpar(0,3,"value")
   spare        = getpar(0,7,"value")
   b1freq       = getpar(0,34,"value")
   time90       = getpar(0,27,"value")
   time180      = getpar(0,5,"value")
   echoTime     = getpar(0,16,"value")
   nrScans      = getpar(0,9,"value")
   recycDelay   = getpar(0,17,"value")

   nrPnts       = acqTime/0.512
   timeRight    = acqTime

   b1txfreq = b1freq
   b1rxfreq = b1freq
```

```
 9852Fword = (( b1txfreq * 2^32 )/125)

 6620Fword = (( b1rxfreq * 2^32 )/62.5)

 echoTime = echoTime - (time90+time180)/2

 gamma = 2.69279e+08

# Save parameters in an array

 p = matrix(32,1)

 p[0] = 0                                #9852 NCO phase etc
 p[1] = (9852Fword & 0xFF000000)/(2^24) #9852 NCO frequency
 p[2] = (9852Fword & 0xFF0000)/(2^16)
 p[3] = (9852Fword & 0xFF00)/(2^8)
 p[4] = (9852Fword & 0xFF)
 p[5] = (6620Fword & 0xFF000000)/(2^24) #6620 NCO frequency
 p[6] = (6620Fword & 0xFF0000)/(2^16)
 p[7] = (6620Fword & 0xFF00)/(2^8)
 p[8] = (6620Fword & 0xFF)
 p[9] = 0 #6620 NCO Phase
 p[10] = 0 #6620 NCO phase
 p[11] = nrPnts               #Number of data points
 p[12] = time90 * 50          #90 Pulse time
 p[13] = time180 * 50         #180 Pulse time
 p[14] = echoTime * 50        #Echo time
 p[15] = recgain * 1.5        #Receiver gain
 p[16] = recycDelay * 1000        #Recycle delay

# Send parameters to DSP

 dspwritepar("x",0x00,p)

# Get display parameters from GUI

 accumulate = getpar(0,25,"text");

# Set number of plot windows

 multiplot("1d",2,1)

# Define matricies for display

 t = [0:1:nrPnts-1]/nrPnts*acqTime;
 f = ([0:1:nrPnts-1]-nrPnts/2)/acqTime;


# Run pulse program and accumlate data

 nrdataPnts = 2*nrPnts
 A = matrix(nrPnts)
 B = matrix(nrPnts)
 E = matrix(nrPnts)
 F = matrix(nrPnts)
 D = matrix(nrdataPnts)
 C = matrix(nrPnts)

 for(n = 1 to nrScans) # Scans loop
```

```
        r = dsprunpp() # Run pulse program
        if(r = 1 | r = 2)
          return # Abort if there's a problem
        endif

        D = dspread("y",0x10000,nrdataPnts)

        for(r = 0 to (nrPnts-2))
              A[r] = D[(2*r)]
              B[r] = D[((2*r)+ 1)]
        next(r)

        if(accumulate = "yes") # Get data
        E = E + A
        F = F + B
        C = sqrt(E^2 + F^2)
        else
        E=A
        F=B
        C = sqrt(A^2 + B^2)
        endif


          drawplot("false")

           curplot("1d",1,1) # Display time domain data
           plot(t,E+i*F)
           #plot(t,C)
           title("text","FID data (Acc: $n$)","size",10)
           xlabel("text","time (us)","size",10)
           ylabel("text","Signal ","size",10)
           axes("fontsize",9)
           zoom1d(timeLeft,timeRight)

           curplot("1d",2,1) # Display frequency domain data
           s = mag(ft(E))
           plot(f,s)
           title("text","Magnitude spectrum (Acc: $n$)","size",10)
           xlabel("text","Frequency (MHz)","size",10)
           ylabel("text","","size",10)
           axes("fontsize",9)

        drawplot("true")

    next(n)

endproc()
```

Appendix C1

A DSP/USB platform for instrumentation and control

R. Dykstra, Proceedings of the 9th Electronics New Zealand Conference, University of Otago, Dunedin, New Zealand, p6-11, 14-15 November, 2002.

A DSP/USB platform for instrumentation and control

R. Dykstra[1]

[1]Institute of Fundamental Sciences
Massey University
Private bag 11-222, Palmerston North, New Zealand.

Email: R.Dykstra@massey.ac.nz

Abstract: A data acquisition, signal processing and control platform has been developed to provide an interface between a computer user and a piece of scientific or industrial equipment. The platform is based around a Motorola DSP, a USB interface and a Windows program. This paper describes the platform and the steps involved in implementing and designing USB devices for Windows-based computers. An outline is given of its current application to Nuclear Magnetic Resonance, and some other potential applications are also suggested.

Keywords NMR, USB, DSP.

1. Introduction

For a number of years Nuclear Magnetic Resonance (NMR) techniques [1] have been used to study the microstructure of Antarctic sea ice. This work consists of inserting a probe into the ice sheet and collecting information about the diffusivity/mobility of the brine trapped within pockets throughout the material and the brine/ice ratio.

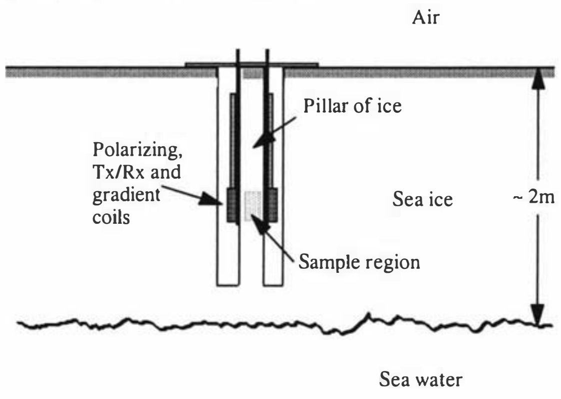


Figure 1. Probing of Antarctic sea ice.

Initially an NMR system that uses the Earth's magnetic field was constructed using a number of mains powered commercial units and some homebuilt electronics and proved the viability of the NMR technique [2,3]. Over time a totally in-house, portable battery powered system has been developed [4] to allow easier access to the ice field and to improve the performance and reliability (figures 2 and 3).

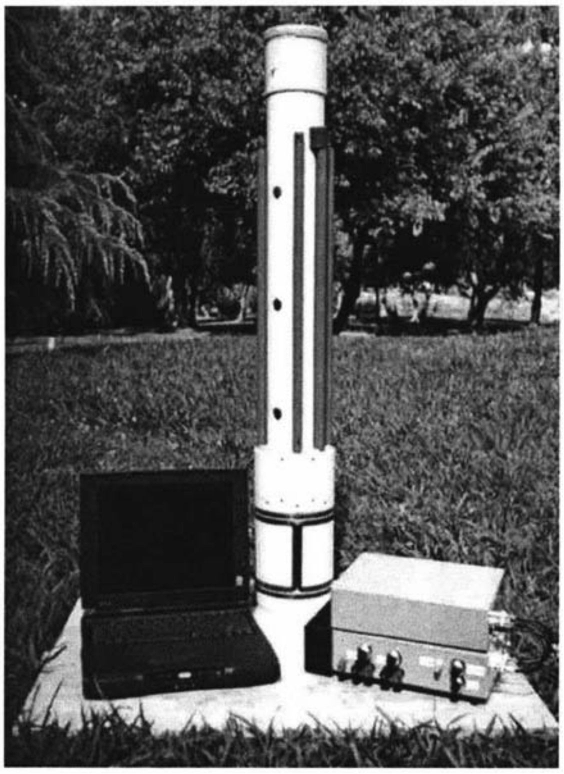


Figure 2. Portable Antarctic NMR system.

NMR like many other techniques requires the stimulation of a sample and then the monitoring of any emissions. It is analogous to the impulse response technique often used to characterise electronic systems.

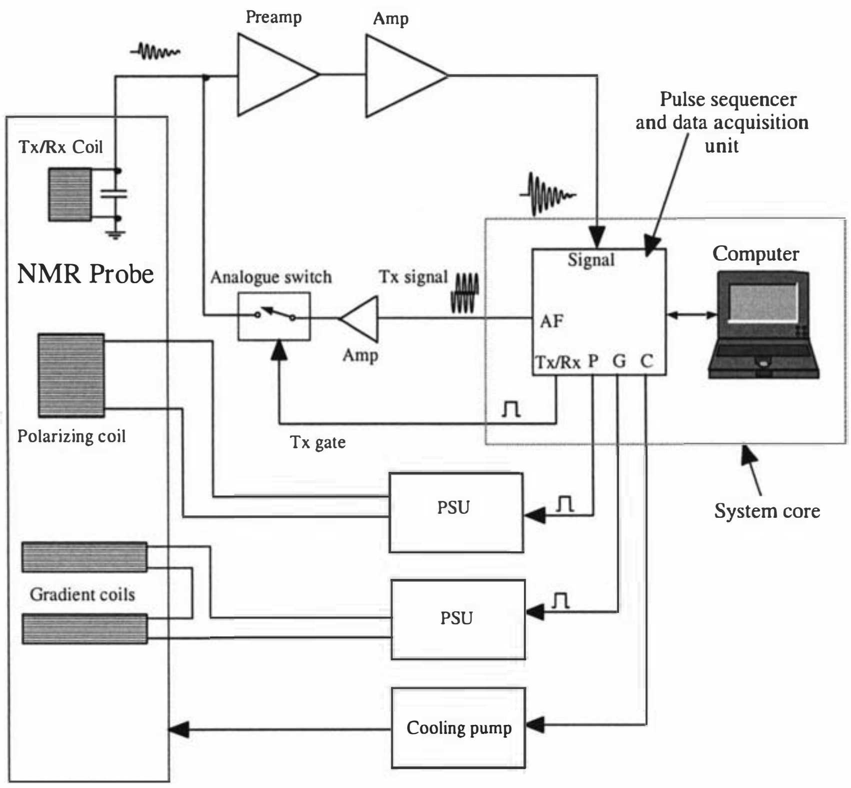


Figure 3. Earth's field NMR system Block diagram.

A radio frequency pulse is applied to a coil surrounding the sample and the subsequent Free Induction Decay (FID) signal is received by the same coil, and after amplification, is acquired. With Earth's Field NMR the stimulus/emission is in the audio frequency region and therefore we can simply use a DAC to generate the transmit pulse and an ADC to acquire the resulting FID. For other high field systems the stimulus/emission is often in the hundreds of MHz region so more complex radio frequency transceiver techniques are required [5]. The timing of the events within the experiments is critical as repeated experiments, coupled with signal averaging, is often used to obtain satisfactory data. Therefore a system is required to provide the necessary control signals and digitise the FID, all with precise timing. This is shown in figure 3 as a block labelled the "System core". Recently some new NMR application areas have been pursued, the main one being the monitoring of the moisture content of construction materials such as concrete. Here a high frequency (20MHz) digital transceiver takes the place of the audio DAC/ADC etc, but a large part, the system core, remains the same. This system core is a common requirement for many other instruments where the generation of control signals and the acquisition of resulting data is required. One example which is also being investigated is Scanning Tunnelling Microscopy. Therefore, a major focus of some recent work has been to develop a general purpose platform that can be used to provide any necessary stimuli, capture resultant emissions and display the data on a computer within a user friendly interface.

Real time control and acquisition is becoming increasingly difficult with recent computers since most modern operating systems are unable to provide guaranteed timing of events and direct accessing of ports and memory spaces. In the past, all one needed to do was to plug in an I/O card and write a few lines of code to access it. Today, one must use trusted device drivers and send I/O requests to these drivers from within the application. The operating system then determines when it will perform the request and this results in massive software overheads that significantly reduce the performance. Therefore in order to meet the timing criteria it is necessary to implement the critical control/acquisition elements with separate hardware. But the major problem still exists of transferring data between the host computer and the system electronics. With plug in cards, RS232 and the Parallel printer port fast becoming redundant, one is now forced to use one of the high speed serial ports such as USB or FireWire. Another area that has increased enormously in complexity, but this time to our advantage, is the user interface. There are a number of pre-packaged software environment providers such as Matlab and Labview that assist in the development of these interfaces, but there are disadvantages in this approach. If a product is being developed that is likely to go into production, a significant cost overhead is software licensing.

2. DSP/USB platform

2.1 *Introduction*

A complete solution from transducer to user that has been developed consists of a DSP board with a full speed USB1.1 interface to communicate with a host computer, a USB Application Programmers Interface (API) and, if required, a complete user interface development environment and data processing package.

2.2 *Platform hardware*

By using a DSP a lot of functional blocks can be implemented in software instead of hardware, and therefore greater flexibility can be achieved. Two examples of this are the real time digital filtering of the NMR FID and the sinusoidal waveform generation using a digital oscillator algorithm. A 24bit,100MHz, MC56309, Motorola DSP was chosen as it has a timing resolution of 10ns, has good dynamic range being a 24bit processor, is available in a 'quad flat pack' surface-mount package and is well supported with a whole host of free development tools such as a C compiler, assembler, simulator and hardware debugger. The MC56309 has 20K x 24bit words of on-chip program (P) memory and two 7K x 24bit words data memories (X and Y).

The additional main components and I/O ports of the board are (figure 4):

- 256K x 24bit high speed memory that can be software configured as extra X,Y or P memory.
- 2 synchronous serial ports that can be used with many ADCs/DACs.
- A hardware debug port.
- 256K x 8bit flash memory that is used to store the operating software which is loaded into the processor upon Reset.
- A full speed USB1.1 interface using the Philips USB D12 device.
- A high speed I/O port consisting of buffered address, control and data lines.

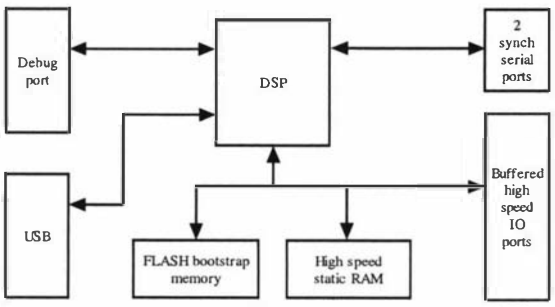


Figure 4. DSP board block diagram.

A standard eurocard sized four layer circuit board is used with a DIN41612 connector at one end for all the I/O lines and by combining with other plug in modules a complete system can be easily built up in a standard rack.

2.3 *Introduction to USB*

USB is a network that uses a tiered star topology to connect multiple devices. This is done with the use of hubs, which are the nodes of the stars within the network. Differential signalling is used to transmit and receive packets between a host controller and a device, in a half duplex arrangement that is capable of transferring data at 12 Mbit/s. The packets are organised into 1 ms duration frames and there are different types of packets that are used to set up transactions, transfer data and provide the necessary handshaking. USB devices can have a number of different endpoints, which are individual buffers that allow parts of the USB device to be uniquely addressed. Using this method multiple transfer types can be in progress simultaneously. Before a data transfer can occur, the host and the device must establish a pipe, which is a logical connection between a device's endpoint and the host controller's software. Each USB device contains a set of descriptors, which are tables that contain information about the device, its configuration and its endpoints. When a device is connected to a USB network, the host controller interrogates the device and obtains the information stored in the descriptors. USB uses four different types of transfers, which are predefined sequences of packets used to define data movement between the controller and a device's endpoint. These are:

- Interrupt. The device cannot initiate any transfers so the controller periodically polls the device to determine if it needs attention.
- Bulk. Is used to transfer large amounts of data.
- Isochronous. Is used to transfer a packet of data every frame.
- Control. Is used for the system control of devices. The controller uses these types of transfers to request devices to perform various standard and vendor defined functions.

The D12 device can be connected to the data bus of a microprocessor and accessed as memory. It consists of a Serial Interface Engine (SIE) that implements the full USB1.1 protocol layer. It is hardwired for speed and needs no firmware intervention as it looks after many functions including: synchronisation pattern recognition, parallel/serial conversion, bit stuffing/de-stuffing, CRC checking/generation, address recognition, and handshake evaluation/generation. The D12 also includes some internal data buffers, which help it to achieve a 1 Mbyte/s data transfer rate for bulk modes. The interface to the host microprocessor is capable of transferring data at the rate of 2 Mbytes/s.

The D12 has three endpoints, a control endpoint that handles all the control requests, a generic endpoint that can be used for interrupt or bulk transfers, and a main endpoint that can be used for isochronous and bulk transfers. The control and generic endpoints both have a maximum packet size of 16 bytes and the main endpoint is limited to a packet size of 64 bytes. The D12 uses a set of commands that a host microprocessor sends to it in order to control its operation. Associated with each command is a data phase, where data is transferred between the D12 and the host microprocessor. Some examples of these commands are: Set Address/Enable, Select Endpoint, Read Buffer and Write Buffer. A single interrupt line is also used by the D12 to signal the host microprocessor when it needs attention, such as when a data packet has been transmitted or received

2.4 *Platform software*

USB development consists of two parts, firstly the design, construction and programming of a USB device, and secondly the development of a device driver and application for the host computer. The platform software supports, besides the control endpoint, two bulk endpoints for the fast transfer of large amounts of data, and two interrupt endpoints for signalling. A 384byte (128 24bit words) data buffer within the internal X memory of the DSP is used as a temporary storage area for bulk in/out transfers.

2.4.1 DSP/USB Board software. The USB/DSP operating system consists of a number of software modules that interact with each other in a layered hierarchical fashion as shown in figure 5.

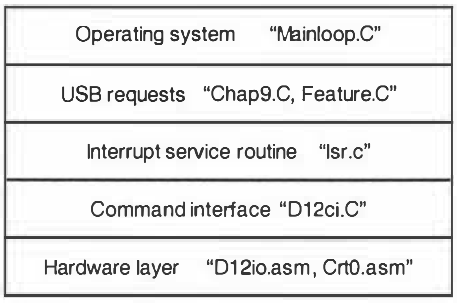

Operating system "Mainloop.C"
USB requests "Chap9.C, Feature.C"
Interrupt service routine "Isr.c"
Command interface "D12ci.C"
Hardware layer "D12io.asm, Crt0.asm"

Figure 5. Layered model of the DSP/USB board software.

The hardware layer is the interface between the hardware and the C part of the operating system. D12io.asm contains some I/O functions that the upper layer (D12ci.C) can call in order to transfer commands and data to the D12. Crt0.asm looks after the system initialisation following a reset and handles the hardware interrupt from the D12 which calls the C interrupt service routine, Isr.C, which is responsible for transferring data between memory buffers and the D12, and for the setting of flags to indicate events to the upper layers.

The module D12ci.C provides a set of command interfaces, that are available to all the layers above it, that encapsulate all the functions used to access the D12. Standard USB requests are handled by the module Chap9.C, and likewise Feature.C handles all the vendor defined requests. The main loop (Mainloop.C) monitors event flags and calls the appropriate subroutines to handle them. For example, when a USB request is received by the interrupt service routine, a flag is set indicating a pending request. The mainloop then detects the pending request, works out if it is a standard or a custom vendor defined request, and then calls the appropriate request handler. Vendor defined requests are a way for USB developers to implement their own features. One example is the "read_write_register" request that is used to send a function request structure to the device that tells it what to perform. Some examples are: setup_bulktransfer, run_pp, get_buffer_size and get_firmware_version.

Using bulk transfers, machine code for the DSP can be loaded into its program memory and then executed through the use of the run_pp request which, after preserving the system stack etc, calls the downloaded code as a subroutine. Upon returning the run_pp request signals the host computer through the interrupt endpoint that it has finished executing the code. Normally the host computer, after requesting run_pp, would poll the interrupt endpoint. The polling itself doesn't interrupt the DSP as it is handled within the D12 device and therefore no problems occur with the program sequence timing. The downloaded code execution can be aborted at any stage by sending another request to the control endpoint. This results in the D12 device signalling an interrupt to the DSP, which then aborts what it was doing and restores its earlier state. This feature of being able to download and execute code has proven to be extremely useful.

2.4.2 Windows software. Windows uses a series of kernel mode (trusted) drivers organised in hierarchical layered fashion as shown in figure 6.

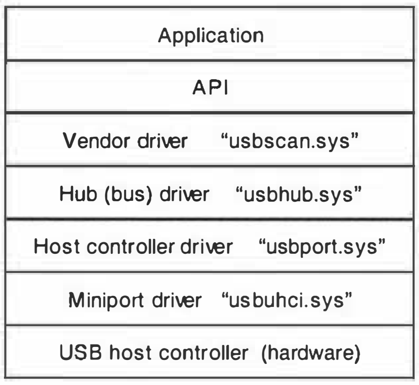

Application
API
Vendor driver "usbscan.sys"
Hub (bus) driver "usbhub.sys"
Host controller driver "usbport.sys"
Miniport driver "usbuhci.sys"
USB host controller (hardware)

Figure 6. Layered model of host computer software.

A custom vendor specific driver is usually required for a USB device, but some standard drivers have been provided that can be used to support particular classes of devices such as scanners, printers and digital cameras. Device drivers are not the easiest things to develop so the platform was designed to mimic a still image class device and therefore the "usbscan.sys" driver provided by Microsoft could be used. Usbscan.sys supports a single control endpoint, along with multiple bulk and interrupt endpoints.

Applications can communicate through the uppermost driver (usbscan.sys) by using a set of standard C functions provided by theWin32 API. Some of these functions that can be called by the application are:

- CreateFile. This is used to obtain a unique identifier (handle) to the device that is used for all subsequent accesses.
- ReadFile. Is used to retrieve data from a device endpoint using bulk transfers.
- WriteFile. Used to send data to a device endpoint using bulk transfers.
- DeviceIOControl. This function together with some predefined I/O control codes and data structures is used to access the control and interrupt endpoints.

The following code segment is an example of how to perform a bulk read transfer:

```
ioRequest.uAddressL = 0x0000; // memory address 010000
ioRequest.bAddressH = 0x01;
ioRequest.uSize = transfer_size;      // n bytes to transfer
ioRequest.bCommand = 0x31;  // read from Y memory

ioBlock.uOffset = 0;
ioBlock.uLength = sizeof(ioRequest);
ioBlock.pbyData = (PUCHAR)&ioRequest;
ioBlock.uIndex = 0x0471;        // bulk transfer request

   result = DeviceIoControl(hdevice,
                    IOCTL_WRITE_REGISTERS,
                    (LPVOID)&ioBlock,
                    sizeof(IO_BLOCK),
                    (LPVOID)NULL,
                    0,
                    (LPDWORD)&nBytes,
                    (LPOVERLAPPED)NULL);

   result = ReadFile(hdevice, rdata, transfer_size, &nBytes,0);
```

The I/O control code: IOCTL_WRITE_REGISTERS is a vendor request that is eventually passed to the USB device, which together with ioBlock.uIndex, is interpreted as a read_write_register request, requesting a bulk transfer. The maximum bulk transfer size is 128 words, so if larger amounts of data need to be transferred, multiple requests are required.

To make it easier for those who want to write their own user interface applications, the Win32 API functions together with all the other necessary information has been combined into a set of higher level, more user friendly functions, shown as the "API" layer in figure 6. Some examples of these higher level functions are:

- DownLoad_program.
- Run_program.
- Abort_program.
- Wait_on_program.
- Read_data.
- Write_data.

The Wait_on_program function is usually called within a separate program thread as it doesn't return until a response is received from the DSP/USB board.

Using these functions means that the application programmer does not need to know anything about USB and all they are concerned with is transferring data etc to/from the USB device's memory. In a sense they communicate with the uppermost layer of the USB system software and all that happens in between is transparent. This is the idea of modern communications where layers are used to hide the complexity from the applications.

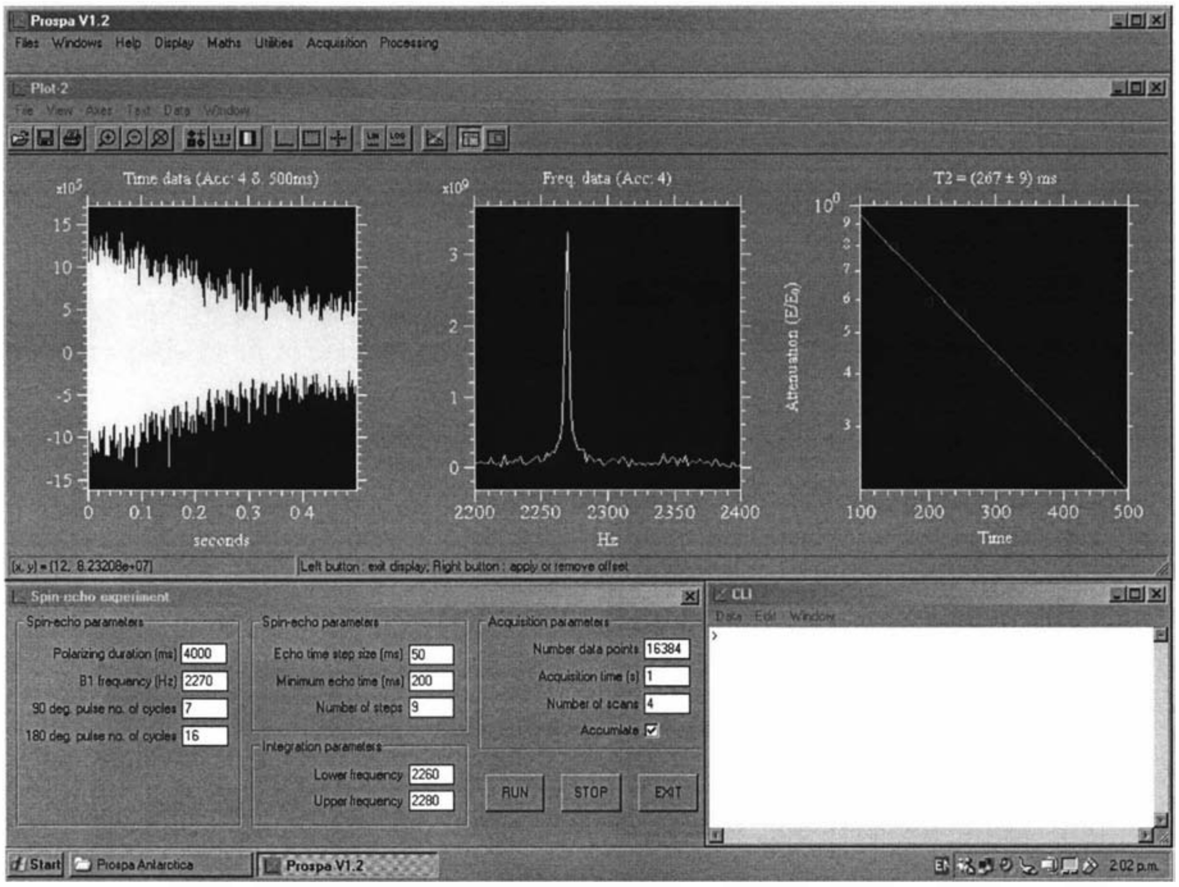
To complete the platform a custom data processing package that includes a scripting language for developing macros and graphical window layouts was developed. Figure 8 shows the user interface for the Antarctic NMR system and demonstrates the capabilities.

3. Conclusion

A platform has been demonstrated that can solve the problem of interfacing real world devices to modern computers and can be used as the basis for many scientific, educational or industrial instruments.

4. Acknowledgements

I wish to thank Prof. Paul Callaghan for his support in this work and Dr Craig Eccles for his helpful suggestions and for developing the user interface application. I also wish to thank the NZ Antarctic program and the NZ Foundation for Research, Science and Technology for logistic and funding support.



(x, y) = (12, 8.23208e+07)

Left button : exit display; Right button : apply or remove offset

Spin-echo experiment

Spin-echo parameters
- Polarizing duration (ms): 4000
- B1 frequency (Hz): 2270
- 90 deg. pulse no. of cycles: 7
- 180 deg. pulse no. of cycles: 16

Spin-echo parameters
- Echo time step size (ms): 50
- Minimum echo time (ms): 200
- Number of steps: 9

Integration parameters
- Lower frequency: 2260
- Upper frequency: 2280

Acquisition parameters
- Number data points: 16384
- Acquisition time (s): 1
- Number of scans: 4
- Accumlate: ✓

RUN STOP EXIT

Figure 8. An example of a user interface

5. References

[1] P. T. Callaghan, *Principals of Nuclear Magnetic Resonance Microscopy*, (1993)

[2] P. T. Callaghan, C. D. Eccles, and J. D. Seymour, Rev. Sci. Instrum., **68**, 4263, (1997)

[3] P. T. Callaghan, R. Dykstra, C. D. Eccles, T. G. Haskell, and J. D. Seymour, Cold Regions Science and Technology, **29**, 153, (1999)

[4] R. Dykstra, *The development of a portable Earth's field NMR system for the study of Antarctic sea ice,* (2000).

[5] C. D. Eccles, C. J. Clark, S. L. Codd and R. Dykstra, Proceedings of the Fifth Electronics New Zealand Conference, 45, (1999).

Appendix C2

Portable NMR systems for non destructive testing

R. Dykstra, P. T. Callaghan, C. D. Eccles and M. W. Hunter, Proceedings of the 2004 New Zealand National Conference on Non Destructive Testing, Massey University, Palmerston North, New Zealand, p166-171, June 27-29, 2004.

R. Dykstra, P. T. Callaghan, C. D. Eccles and M. W. Hunter, Journal of the Australian Institute for non-destructive testing, Vol. 42, No.1, p15-18, January/February, 2005.

R. Dykstra, P. T. Callaghan, C. D. Eccles and M. W. Hunter, Journal of the Australian Institute for non-destructive testing, Vol. 42, No.4, p114-117, July/August, 2005.

PORTABLE NMR SYSTEMS FOR NON DESTRUCTIVE TESTING

R. Dykstra*, P. T. Callaghan**, C. D. Eccles* and M. W. Hunter**

* Institute of Fundamental Sciences, Massey University, Palmerston North, New Zealand
R.Dykstra@massey.ac.nz

** MacDiarmid Institute of Advanced Materials and Nanotechnology, School of Chemical and Physical Sciences, Victoria University of Wellington, Wellington, New Zealand

Abstract

Nuclear Magnetic Resonance (NMR) is a relatively complex technique and normally requires expensive equipment, however with advances in computing, electronics and permanent magnet technologies, NMR is becoming more feasible as a non-invasive tool for industry. The strength of NMR is its ability to probe at the molecular level and hence gain information about molecular structure, organisation, abundance and orientation. This paper presents some of the work being undertaken in developing low cost portable NMR systems for the non-destructive testing of materials such as polymer composites, rubber, timber and concrete.

1. INTRODUCTION

Nuclear Magnetic Resonance (NMR) is one of the more recent sensing technologies and has become very popular for its ability to non-invasively probe down to the molecular level the properties of many materials and living organisms. Its greatest impact has been in the areas of chemistry and medical radiology, but now it is being applied to biochemistry, structural biology, and materials science research [1]. In the past ten years, NMR has made significant contributions to horticulture [2,3], biotechnology, chemical engineering, petroleum science and food technology and now stands on the threshold of making an impact on environmental monitoring, building technology, and security technology. Traditionally NMR/MRI is performed using laboratory or clinic based superconducting magnets, but now it is moving out into industry in the form of portable permanent magnet based systems. NMR development is being driven by advancements in electronics, computing and magnet technology and so continues to advance in capability and application.

In the building industry, it is often necessary to determine the moisture content of concrete so that it can be optimally cured for strength or to know when floor coverings can be applied. Presently the only way to accurately gauge the true moisture content is to use destructive techniques. This is the method that Parrot [4] used to generate the graph shown in figure 1. Typically the industry yardstick of a "month per inch" is used for estimating the time required before the concrete surface can be covered, but from this data it is clear that even after a year there can still remain a significant amount of moisture.

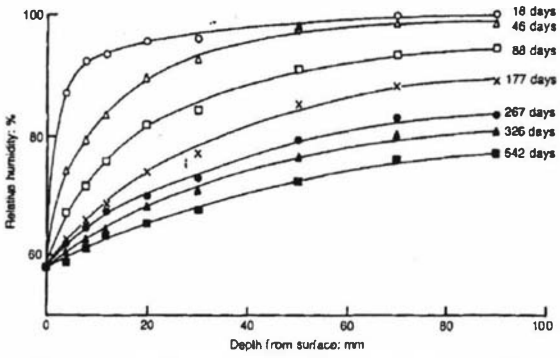


Figure 1. Moisture profiles of concrete for various drying times [4].

Concrete is inherently porous and so is greatly affected by the environment and the laying process methods. This makes it very difficult to make predictions based on calibrated standards. What is required is a portable instrument that can be taken to a site to non-destructively obtain the moisture content. NMR technology is now being developed for this specific application in the form of a permanent magnet based probe and an associated backpack containing the necessary signal processing, control electronics and batteries.

2. NUCLEAR MAGNETIC RESONANCE

Certain nuclei possess an intrinsic angular momentum and magnetic moment and when placed in a magnetic field, B_0, the nuclear moments will precess about the applied field direction at the Larmor frequency $\omega_0 = \gamma B_0$. The spins may be aligned or opposed to the field direction. At room temperature more spins are aligned than opposed with the field giving rise to a net magnetization vector, **M**, aligned with the applied field.

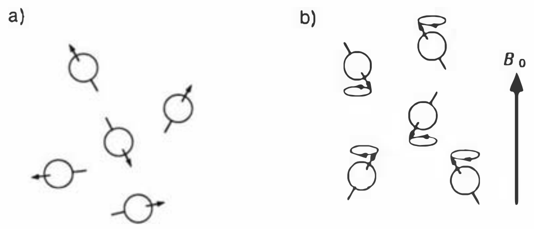


Figure 2. Nuclear magnetic moments a) before and b) after the application of a magnetic field.

We interact with this magnetization by applying short radio frequency (RF) pulses at the Larmor frequency to a resonator surrounding the sample. The magnetic component of this pulse, B_1, causes the magnetization to rotate away from the z (B_0) axis into the x-y plane. Once the pulse has finished the magnetization will continue to precess about B_0 at ω_0. The same resonator which produced the B_1 pulse can now be used to detect the EMF induced by the precessing magnetization. This induced signal, called the free-induction decay or FID, has a level ranging from μV to mV and takes the form of a decaying sine wave. The decay arises from relaxation processes which cause the perturbed magnetization to return to its equilibrium position.

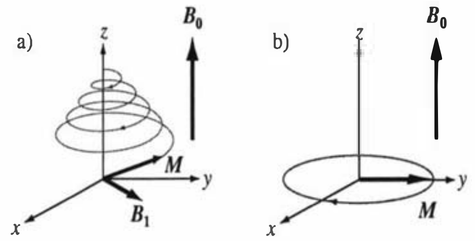


Figure 3. a) During the B_1 pulse the magnetization spirals down about the z axis until it ends up in the x-y plane. b) Following the B_1 pulse the magnetization precesses freely in the x-y plane at frequency ω_0.

Superconducting magnets have been the key to the success of laboratory based NMR systems, however they typically cost millions of dollars to purchase, are also very expensive to maintain and require special facilities to house them. But the information that NMR can obtain makes them invaluable.

Figure 4. A 14 Tesla superconducting magnet of height approximately 2 m [5].

Nuclei are affected by other surrounding nuclei which gives rise to a distribution of resonant frequencies. This property is the heart of NMR spectroscopy and allows the determination of nuclear structure. By applying additional magnetic field gradients, spatial information can also be obtained. This is the basis for Magnetic Resonance Imaging (MRI).

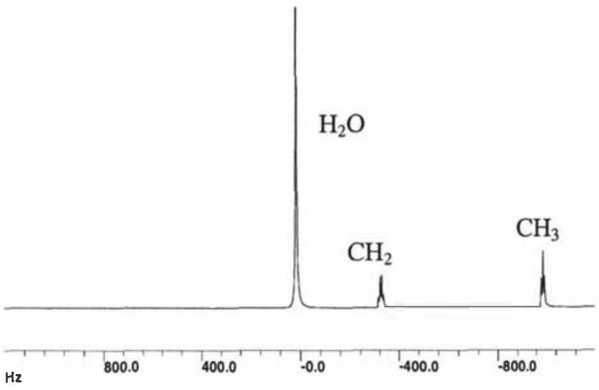


Figure 5. Spectrum of a Tequila sample.

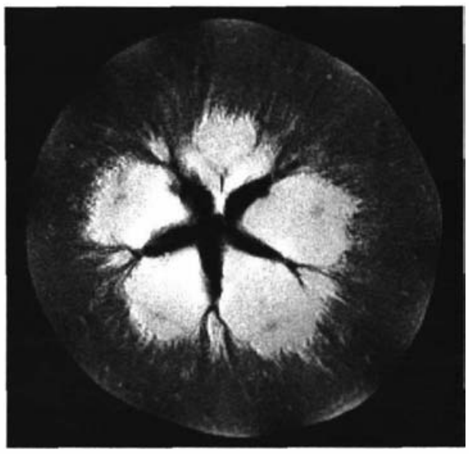


Figure 6. An apple suffering from water core, a condition which shows improvement on storage .

3. PORTABLE NMR

NMR is now moving out of the laboratory and into the field in the form of low cost permanent magnet based systems. They are very limited in performance when compared to laboratory systems but nevertheless are still very useful tools for specific applications where only minimal amounts of information are required.

3.1 The NMR "MOUSE"

One example of such a probe is the **M**obile **U**niversal **S**urface **E**xplorer (MOUSE) [6]. It was initially developed for analysing the rubber on car tyres, but has now been applied to areas such as the moisture determination of historic documents and buildings. The device is based on the principles of "inside-out" NMR where the region of interest is external to the probe. The NMR-MOUSE is a surface probe that can only be used to obtain data from samples that are located within 1 mm of the probe surface. The NMR-MOUSE is relatively easy to construct and is made from readily available materials that cost in the order of a few hundred dollars.

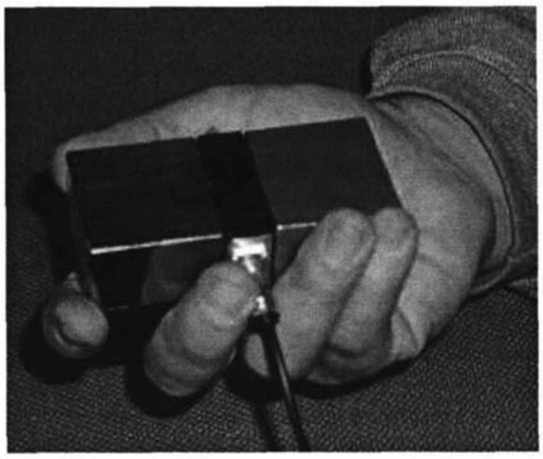


Figure 7. The NMR-MOUSE [6].

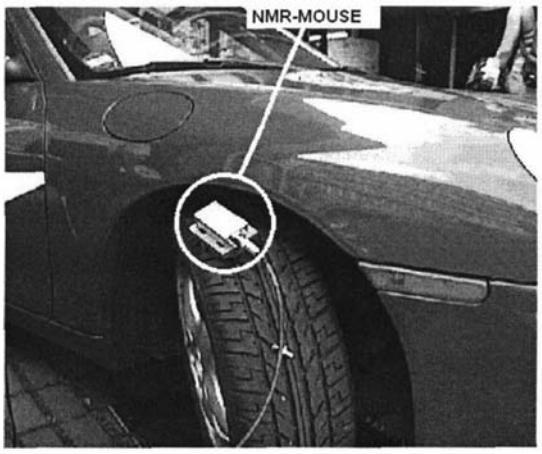


Figure 8. Using the NMR-MOUSE to monitor tyre rubber.

3.2 The NMR "MOLE"

What is required for applications such as the monitoring of drying concrete is a probe that can obtain data from deep inside the material. The challenge that is facing designers of "inside-out" portable systems is the need to generate a region of homogeneous field remote from the surface of the probe. Our aim for a new probe called the NMR-MOLE was to produce a one cubic centimetre region 50 mm into the sample. This depth would be sufficient to enable the monitoring of drying concrete. It was achieved using a series of individual magnets to approximate a ring magnet and a further axial magnet in the center. The required field was defined in terms of its Taylor coefficients, in this case the first and second being zero. The position of central magnet and the axial angle of the ring magnets were adjusted using Newton's method until the desired coefficients were found at the prescribed depth. This proved to be a simple and robust technique to solve the problem of generating the required field. The magnetic field calculations involved numerical simulations using a Biot-Savart calculation in MATLAB [7]. The array was constructed and the design verified using field mapping techniques. A region 55 mm deep, of field strength 10 mT with a volume of approximately $10^{-6}m^3$ was achieved. A simple B_1 surface coil was designed and built to try and optimize signal-to-noise.

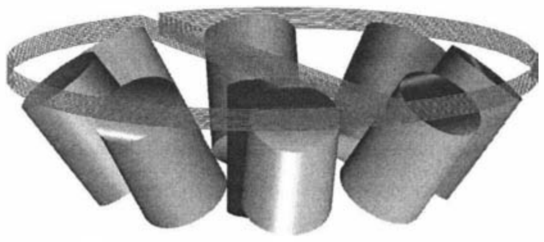


Figure 9. The configuration of magnets used to produce a homogenous region 55 mm into the sample surface. The B_1 coil is also shown [8].

3.3 System electronics

3.3.1 System Core

What is common to all NMR applications is the generation of precisely timed signals, the capturing of FIDs and the processing/display of data. Most of this we have encapsulated into a single unit known as the system core [9]. This is based around a general purpose Digital Signal Processor (DSP) and a Universal Serial Bus (USB) interface that is used to communicate with a host laptop computer. A graphical user interface is provided by an application running on Windows XP. This interface is fully configurable and scriptable through the use of a macro language and a set of standard commands. Pulse programs can be generated using the built-in language provided by the user interface or by using a C compiler/assembler to generate code for the DSP itself. The DSP runs at 100 MHz and therefore provides a pulse program timing resolution of 10 ns. The system core has also been used for non-NMR applications, one example being a magnetic field mapper used to verify magnet designs.

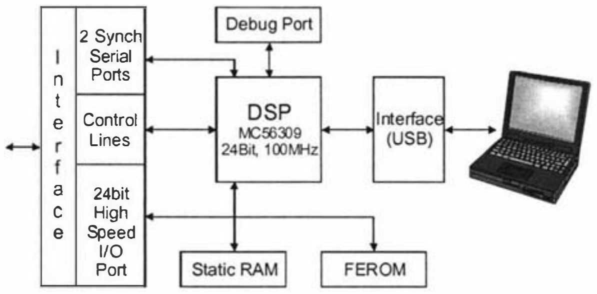


Figure 10. System core.

3.3.2 Digital Receiver

The RF section of an NMR system is very similar to a communications transceiver, therefore we can take advantage of technology advances in this area. A major advancement in recent years has been the introduction of digital transceiver technology. One example is the AD6620 from Analog Devices [10] which is shown in figure 10.

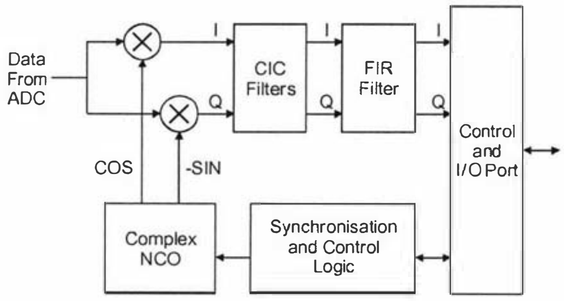


Figure 11. The AD6620.

Here the received signal is sampled immediately after the IF stage or the preamplifier. It is then mixed digitally with synthesized sine and cosine functions to generate the lower frequency quadrature outputs. Digital filtering is then applied to reduce the bandwidth and finally down-sampling is used to reduce the output data rate. This technique guarantees gain matching and zero DC offsets of the quadrature channels. This eliminates the need for the phase cycling techniques that are usually required to overcome these problems. A digital receiver was developed based on the AD6620. A block diagram is shown below.

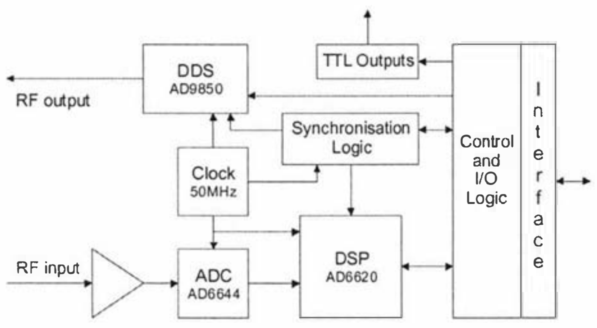


Figure 12. Digital transceiver.

The AD6644 14-bit ADC uses a constant sampling rate of 50 MHz and sends its digital values directly to the AD6620. The output bandwidth of the AD6620 receiver can be up to 1 MHz and its output is fed to the system core DSP for further processing and storage. A Direct Digital Synthesizer (DDS) is also included to generate any required B_1 signals. Both the synthesizer and the AD6620 NCO have phase and frequency hopping capabilities and are phase locked to each other and the DSP. The frequencies are determined by the user and can be set between 0 and 25 MHz with a frequency resolution of less than 0.02 Hz. The analogue input to the receiver has a 250 MHz bandwidth allowing under-sampling techniques to be used. The DSP/USB unit and RF transceiver are each built on standard Euro-card sized multi-layer PCBs (figure 12) and connect to each other via a back plane. This modular approach enables a system to be built using a standard rack mount case.

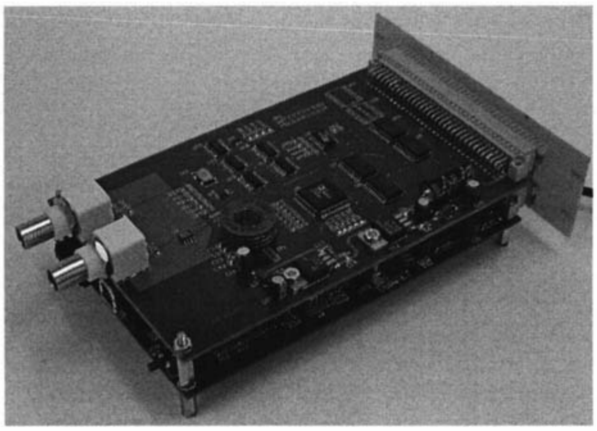


Figure 13. DSP/USB unit and digital transceiver.

4. CONCLUSION

A partially completed NMR system consisting of the NMR MOLE probe and the NMR platform has been designed and constructed and is shown in Figure 13. The next stage is to produce a series of RF power amplifiers, preamplifiers and duplexors that can be used with the MOLE and other probes for a range of applications. Shortly the NMR platform will be combined with the NMR-MOUSE to produce a complete portable system for studying the properties of various surface materials. This system will take the form of a backpack containing all the electronics and batteries and the user interface will be provided through the use of a handheld computer. The NMR platform itself can be used as the heart of many scientific, educational or industrial instruments.

5. ACKNOWLEDGEMENTS

The authors would to thank the technical staff from the science workshops of Massey University and Victoria University of Wellington. This project was made possible by funding provided by the Foundation for Research Science and Technology.

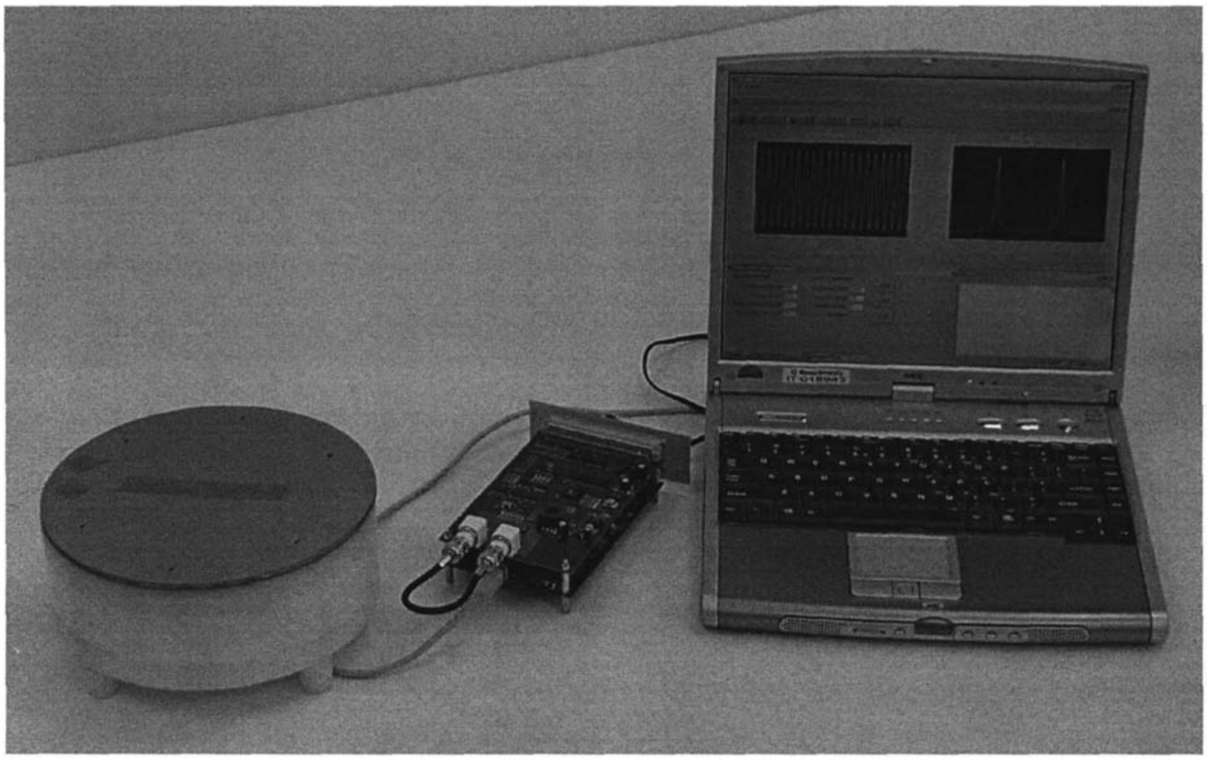


Figure 14. Partially completed system .

6. REFERENCES

[1] Callaghan, P.T., "Principles of Nuclear Magnetic Resonance Microscopy", Oxford University Press, (1991).

[2] Eccles, C.D., Clark, C.J., Codd, S.L. and Dykstra, R., "Construction of an MRI system for Horticultural and Materials Science Research", Proceedings of the Fifth Electronics New Zealand Conference, 1998, pp 45-50.

[3] Hills, B.P. and Clark, C.J., "Quality Assessment of Horticultural Products by NMR", Ann Report NMR Spect, 50, 2003, pp 76-117.

[4] Parrot, L.J., Magazine of concrete research, 1991, 43, No. 154, pp 45-52.

[5] www.bruker.com

[6] Eidmann, G., Savelsberg, R., Blumler, P. and Blumich B., "The NMR MOUSE, a Mobile Universal Surface Explorer", J. Magn. Reson, A **122**, (1996), pp 104-109.

[7] www.mathworks.com

[8] NMR Apparatus. New Zealand patent application 520114 / International Patent Application No. PCT/ NZ03/00149

[9] Dykstra, R., "A DSP/USB platform for instrumentation and control", Proceedings of the Ninth Electronics New Zealand Conference, 2002, pp 6-11.

[10] www.analog.com

Appendix C3

The Development of a Portable NMR System

R. Dykstra, M. Adams, V. Anferov, B. Blumich, P. T. Callaghan, C. D. Eccles and M. W. Hunter, Proceedings of the 11th Electronics New Zealand Conference, Massey University, Palmerston North, New Zealand, p49-54, 15-16 November, 2004.

The Development of a Portable NMR System

R. Dykstra[1], M. Adams[3], V. Anferov[3], B. Blumich[3], P. T. Callaghan[2], C. D. Eccles[2] and M. W. Hunter[2]

[1] Institute of Fundamental Sciences, Massey University, Palmerston North, New Zealand

Email: R.Dykstra@massey.ac.nz

[2] MacDiarmid Institute of Advanced Materials and Nanotechnology, School of Chemical and Physical Sciences, Victoria University of Wellington, Wellington, New Zealand.

[3] Institute for Technical Chemistry and Macromolecular Chemistry, Aachen University of Technology RWTH, D-52056, Aachen, Germany.

Abstract: Nuclear Magnetic Resonance (NMR) is a relatively complex technique and normally requires expensive equipment, however with advances in computing, electronics and permanent magnet technologies, NMR is becoming more feasible as a non-invasive tool for industry. The strength of NMR is its ability to probe at the molecular level and hence gain information about molecular structure, organisation, abundance and orientation. This paper presents some of the work being undertaken in developing low cost portable NMR systems for the non-destructive testing of materials such as polymer composites, rubber, timber and concrete.

Keywords Nuclear Magnetic Resonance, Non Destructive Testing, Permanent magnets

1. INTRODUCTION

Nuclear Magnetic Resonance (NMR) is one of the more recent sensing technologies and has become very popular for its ability to non-invasively probe down to the molecular level the properties of many materials and living organisms. Its greatest impact has been in the areas of chemistry and medical radiology, but now it is being applied to biochemistry, structural biology, and materials science research [1]. In the past ten years, NMR has made significant contributions to horticulture [2,3], biotechnology, chemical engineering, petroleum science and food technology and now stands on the threshold of making an impact on environmental monitoring, building technology, and security technology. Traditionally NMR/MRI is performed using laboratory or clinic based superconducting magnets, but now it is moving out into industry in the form of portable permanent magnet based systems. NMR development is being driven by advancements in electronics, computing and magnet technology and so continues to advance in capability and application.

In the building industry, it is often necessary to determine the moisture content of concrete so that it can be optimally cured for strength or to know when floor coverings can be applied. Presently the only way to accurately gauge the true moisture content is to use destructive techniques [4]. Typically the industry yardstick of a "month per inch" is used for estimating the time required before the concrete surface can be covered, but it is known that even after a year there can still remain a significant amount of moisture. Concrete is inherently porous and so is greatly affected by the environment and the laying process. This makes it very difficult to make predictions based on calibrated standards. What is required is a portable instrument that can be taken to a site to non-destructively obtain the moisture content.

Portable NMR technology has now been developed for this specific application and other applications such as the monitoring of rubber, polymers, timber and many other materials. This new technology will lead to a range of systems consisting of a permanent magnet based probe and an associated equipment case or backpack containing the necessary signal processing, control electronics and batteries.

2. NUCLEAR MAGNETIC RESONANCE

Certain nuclei possess an intrinsic angular momentum and magnetic moment and when placed in a magnetic field, B_0, the nuclear moments will precess about the applied field direction at the Larmor frequency $\omega_0 = \gamma B_0$. The spins may be aligned or opposed to the field direction. At room temperature more spins are aligned than opposed with the field giving rise to a net magnetization vector, M, aligned with the applied field.

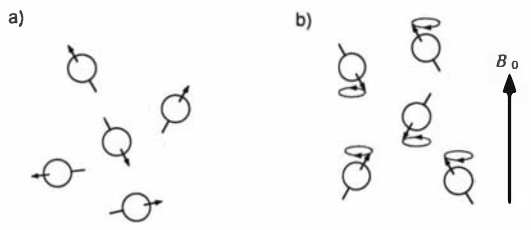


Figure 1. Nuclear magnetic moments a) before and b) after the application of a magnetic field.

We interact with this magnetization by applying short radio frequency (RF) pulses at the Larmor frequency to a resonator surrounding the sample. The magnetic component of this pulse, B_1, causes the magnetization to rotate away from the z (B_0) axis into the x-y plane. Once the pulse has finished the magnetization will continue to precess about B_0 at ω_0. The same resonator which produced the B_1 pulse can now be used to detect the EMF induced by the precessing magnetization. This induced signal, called the free-induction decay or FID, has a level ranging from µV to mV and takes the form of a decaying sine wave. The decay arises from relaxation processes which cause the perturbed magnetization to return to its equilibrium position.

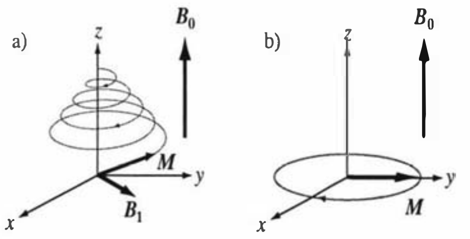


Figure 2. a) During the B_1 pulse the magnetization spirals down about the z axis until it ends up in the x-y plane. b) Following the B_1 pulse the magnetization precesses freely in the x-y plane at frequency ω_0.

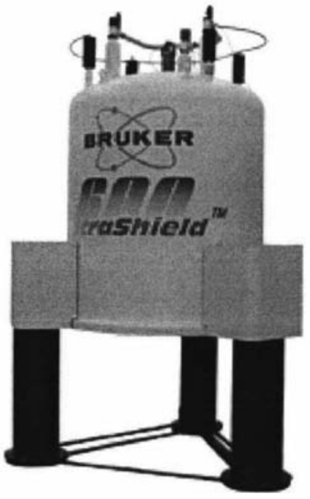


Figure 3. A 14 Tesla superconducting magnet of height approximately 2 m [5].

Superconducting magnets have been the key to the success of laboratory based NMR systems, however they typically cost millions of dollars to purchase, are also very expensive to maintain and require special facilities to house them. But the information that NMR can obtain makes them invaluable. Nuclei are affected by other surrounding nuclei which gives rise to a distribution of resonant frequencies. This property is the heart of NMR spectroscopy and allows the determination of nuclear structure. By applying additional magnetic field gradients, spatial information can also be obtained. This is the basis for Magnetic Resonance Imaging (MRI).

3. PORTABLE NMR SYSTEMS

NMR is now moving out of the laboratory and into the field in the form of low cost permanent magnet based systems. They are limited in performance when compared to laboratory systems but nevertheless are still very useful tools for specific applications where lesser amounts of information are required.

Every NMR system whether laboratory based or portable basically consists of a probe, a whole set of supporting electronics and a user interface for controlling the system and observing/collecting the data. This is pictured below in figure 4.

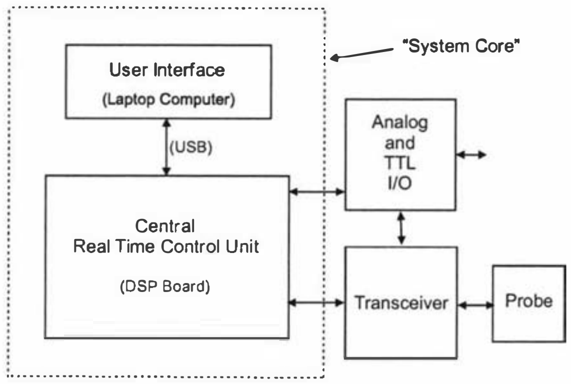


Figure 4. NMR System Architecture

3.1 Portable NMR probes

The probe is the sensor and can come in a range of sizes and designs. This is the critical interface between the material that one wants to observe and the user and therefore determines the performance of the whole system.

3.1.1 The NMR "MOUSE"

One example of a portable probe is the **Mo**bile **U**niversal **S**urface **E**xplorer (MOUSE) developed by a group of researchers within the Aachen University of Technology [6]. It was initially developed for analysing the rubber on car tyres, but has now been applied to areas such as the moisture determination of historic documents and buildings. The device is based on the principles of "inside-out" NMR where the region of interest is external to the probe. The NMR-MOUSE is a surface probe that can only be used to obtain data from samples that are located within a few mm of the probe surface. The NMR-MOUSE is relatively easy to construct and is made from readily available materials that cost in the order of a few hundred dollars. Figure 5 gives a schematic view of how the probe is constructed and figure 6 shows the completed hand held unit. Two rectangular permanent magnets are placed on an iron yoke to generate the necessary B_0 field that is aligned with the surface of the probe. A solenoidal B_1 coil is used to interact with any sample that is placed near the probe surface.

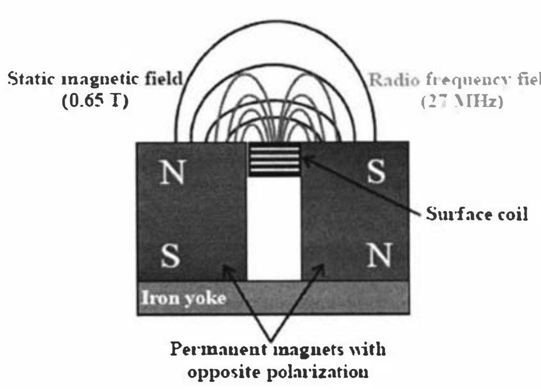


Figure 5. NMR-MOUSE, magnet and B_1 coil configuration [6].

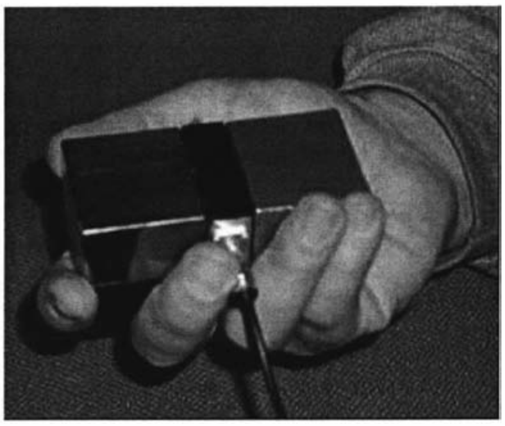


Figure 6. Two magnet, handheld NMR-MOUSE.

An alternative to this design that again the people in Aachen have produced is the single cylindrical bar magnet mouse shown in figure 7. Here a B_1 coil is placed directly on the surface of a magnet and again is designed to generate a field that is perpendicular to the B_0 field of the magnet, but this time parallel with the surface of the probe. This is the opposite to the configuration of the earlier two magnet MOUSE design. Again only samples within a few mm from the surface can be measured.

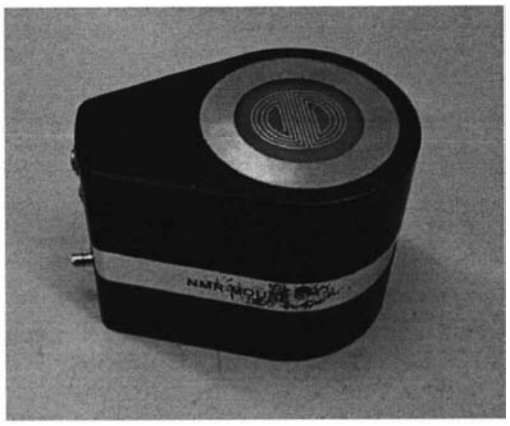


Figure 7. The single bar magnet NMR-MOUSE [6].

3.1.2 The NMR "MOLE"

What is required for applications such as the monitoring of drying concrete is a probe that can obtain data from deep inside the material. The challenge that is facing designers of "inside-out" portable systems is the need to generate a region of homogeneous field remote from the surface of the probe. The aim for a new probe called the NMR MOLE was to produce a one cubic centimetre region 50 mm into the sample. This depth would be sufficient to enable the monitoring of drying concrete. It was achieved using a series of individual magnets to approximate a ring magnet and a further axial magnet in the centre. The position of central magnet and the axial angle of the ring magnets were adjusted until the desired field was obtained. The array was constructed and the design verified using field mapping techniques. A region 55 mm deep, of field strength 10 mT with a volume of approximately 10^{-6} m^3 was achieved. The probe has a diameter of 250 mm and weighs approximately 6 kg.

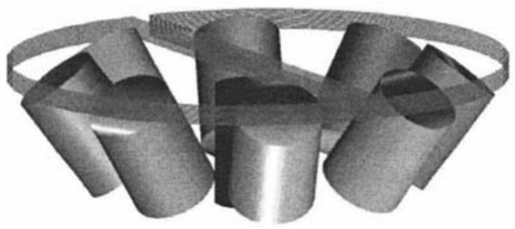


Figure 8. The configuration of magnets used to produce a homogenous region 55 mm into the sample surface [7]. The B_1 coil is also shown.

3.2 System electronics

NMR like many other techniques requires the stimulation of a sample and then the monitoring of any emissions. It is analogous to the impulse response technique often used to characterise electronic systems. A radio frequency pulse is applied to a coil close to the sample and the subsequent Free Induction Decay (FID) signal is received by the same coil, and after amplification, is acquired. For permanent magnet systems the stimulus/emission is often in the tens of MHz region so radio frequency transceiver techniques are required. The timing of the events within the experiments is critical as repeated experiments, coupled with signal averaging, are often used to obtain satisfactory data.

3.2.1 System Core

What is common to all NMR applications is the generation of precisely timed signals, the capturing of FIDs and the processing/display of data. Most of this has been encapsulated into a single unit known as the system core [8]. This is based around a general purpose Digital Signal Processor (DSP) and a Universal Serial Bus (USB) interface that is used to communicate with a host laptop computer. A graphical user interface is provided by an application running on Windows XP. This interface is fully configurable and scriptable through the use of a macro language and a set of standard commands. Pulse programs can be generated using the built-in language provided by the user interface or by using a C compiler/assembler to generate code for the DSP itself. The DSP runs at 100 MHz and therefore provides a pulse program timing resolution of 10 ns. The system core has also been used for non-NMR applications, one example being a magnetic field mapper used to verify magnet designs. A block diagram is shown below.

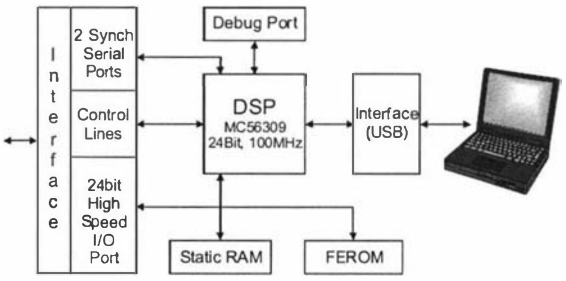


Figure 9. The system core consisting of a DSP board that uses USB to interface to a host Laptop computer.

3.2.2 Digital Transceiver

The RF section of an NMR system is very similar to a communications transceiver, therefore we can take advantage of technology advances in this area. A major advancement in recent years has been the introduction of digital transceiver technology. One example is the AD6620 from Analog Devices [9] which is shown in figure 10. Here the received signal is sampled immediately after the IF stage or the preamplifier. It is then mixed digitally with synthesized sine and cosine functions to generate the lower frequency quadrature outputs. Digital filtering is then applied to reduce the bandwidth and finally down-sampling is used to reduce the output data rate. This technique guarantees gain matching and zero DC offsets of the quadrature channels. This eliminates the need for the phase cycling techniques that are usually required to overcome these problems. A digital transceiver was developed based on the AD6620. A block diagram is shown in figure 11.

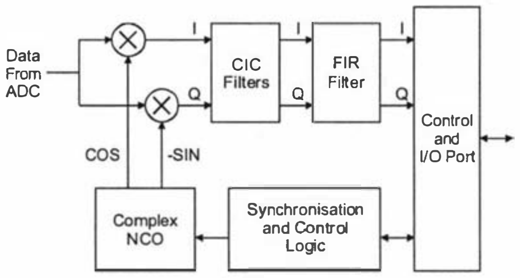


Figure 10. The AD6620.

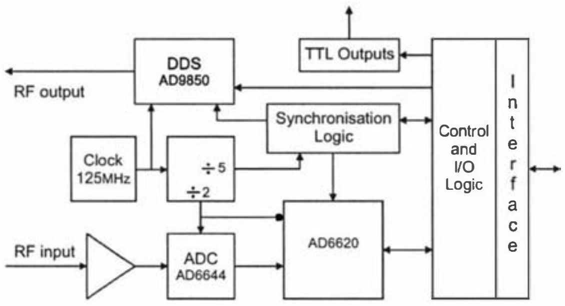


Figure 11. Digital transceiver.

The AD6644 14-bit ADC uses a constant sampling rate of 62.5 MHz and sends its digital values directly to the AD6620. The output bandwidth of the AD6620 receiver can be up to 1 MHz and its output is fed to the system core DSP for further processing and storage. An AD9850 Direct Digital Synthesizer (DDS) is also included to generate any required B_1 signals. Both the synthesizer and the AD6620 NCO have phase and frequency hopping capabilities and are phase locked to each other and the DSP. The frequencies are determined by the user and can be set between 0 and 30 MHz with a frequency resolution of less than 0.02 Hz. The DSP/USB unit and RF transceiver are each built on standard Euro-card sized multi-layer PCBs (figure 12) and connect to each other via a back plane.

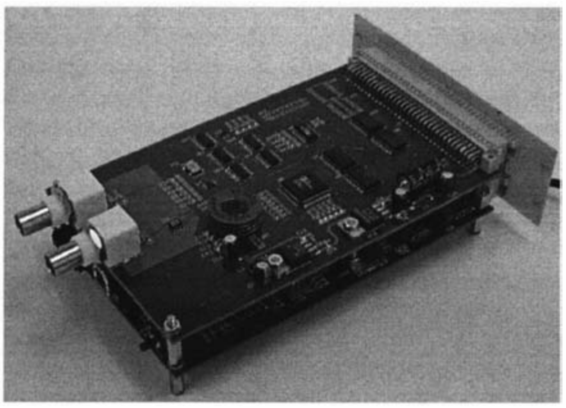


Figure 12. DSP/USB unit and digital transceiver.

3.2.3 *RF electronics*

To complete the NMR spectrometer system a RF front end is still required. What is needed here is a 250 W, 20 MHz broadband linear RF pulse amplifier, a very sensitive low noise preamp and a fast transmit/receive switch. Figure 13 below shows the arrangement that is used.

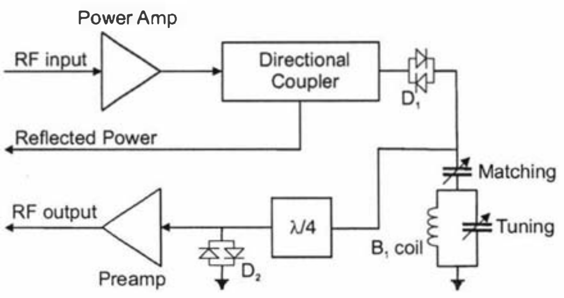


Figure 13. RF front end.

A directional coupler is used for tuning and matching the probe coil to 50 Ω, diodes D1 are used to block any residual noise from the power amp during receive and diodes D2 together with a lumped element quarter wavelength transmission line act as a transmit/receive switch to protect the preamp during transmission. During transmission all the diodes conduct due to the high voltages involved and therefore diodes D2 present themselves as a short circuit to the quarter wavelength transmission line which in turn appears as a high impedance to the RF power source resulting in minimal power going into the very delicate preamp. Typically the transmit pulse duration ranges between 2 and 10 μs and can be repeated every 30 μs.

3.2.4 *Complete Portable NMR spectrometer*

The complete NMR spectrometer consisting of the DSP/USB unit, the digital transceiver and RF front end was housed in a single aluminium case of dimensions 300 x 200 x 50 mm as shown in figure 14. The system requires a single 24 VDC, 2 A supply.

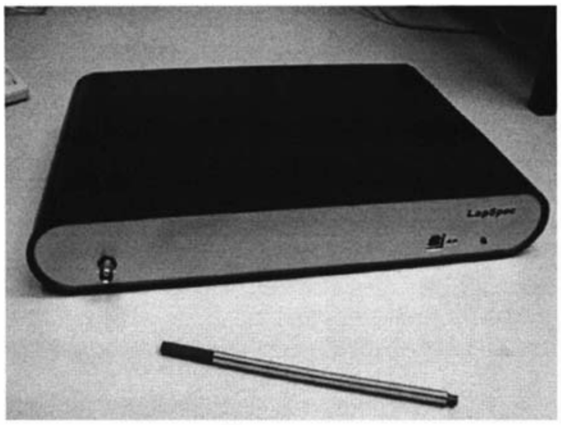
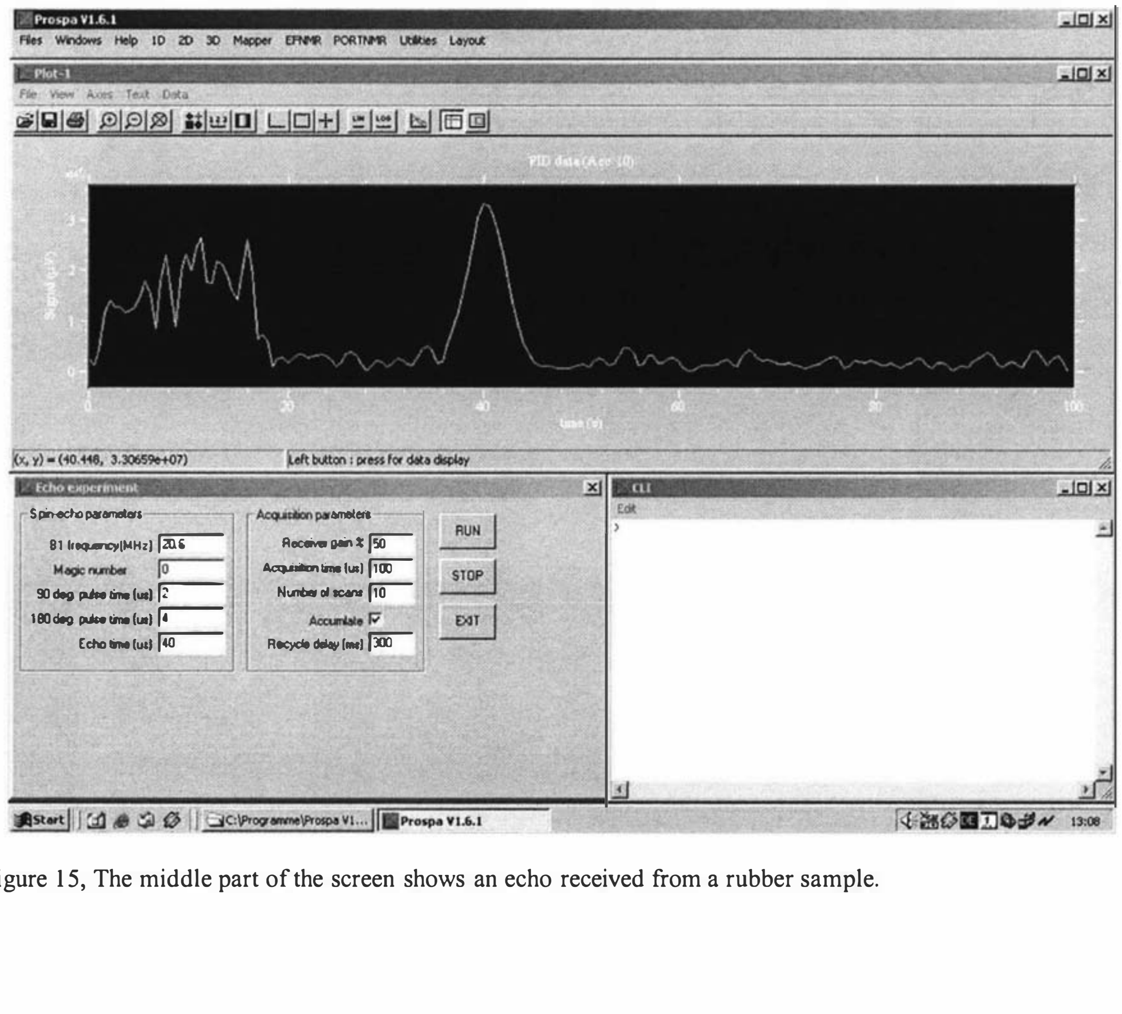


Figure 14. Complete portable NMR spectrometer.



(x, y) = (40.448, 3.30659e+07)
Left button : press for data display

Echo experiment

Spin-echo parameters

B1 frequency(MHz)	20.6
Magic number	0
90 deg pulse time (us)	2
180 deg pulse time (us)	4
Echo time (us)	40

Acquisition parameters

Receiver gain %	50
Acquisition time (us)	100
Number of scans	10
Accumulate	✓
Recycle delay (ms)	300

Figure 15, The middle part of the screen shows an echo received from a rubber sample.

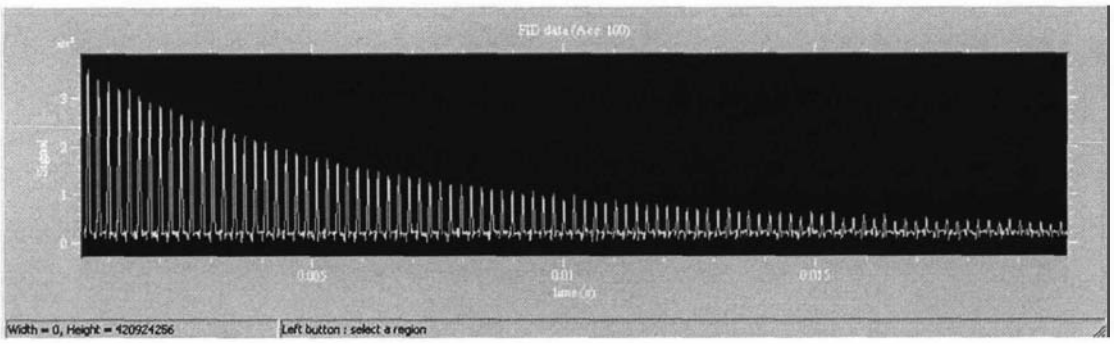


Figure 16, CPMG experiment, where multiple refocusing pulses are applied to produce a series of echos.

The spectrometer was used with a single bar magnet MOUSE and a rubber sample to verify that the whole system was operating correctly. Figure 15 shows an echo signal obtained using 10 experiment averages. Some time after the first RF pulse, another RF pulse is applied to refocus the dephasing magnetization to form an echo. The amplitude of the echo is directly proportional to the number of protons in the sample and therefore for some samples is directly proportional to the moisture content. This refocusing idea can be repeated many times so that a series of echos can be obtained and this is shown in figure 16. This is called a Carr-Purcell-Meiboom-Gill, (CPMG) sequence and is used to obtain information about relaxation processes that are occuring within the sample material. The decay can tell us about the molecular environment of the protons and therefore give us information about some of the material properties. Figure 14 also shows the user interface for the system. The user can design their own control panels and write macros to control the functionality of the system and the processing of any data.

4. CONCLUSION

A completed NMR system consisting of the NMR-MOUSE probe and the NMR spectrometer has been designed, constructed and verified. Some more work is still required to fine tune the system for optimum performance. The next stage is to combine it with the NMR-MOLE for the concrete application and other probes for a range of applications. Some additional work will be designing a programming language for NMR experiment developers, implementation of some additional digital filtering and continuing to further reduce the size. Eventually the system may take the form of a backpack containing all the electronics and batteries and the user interface will be provided through the use of a handheld computer. The NMR platform itself can be used as the heart of many scientific, educational or industrial instruments.

5. ACKNOWLEDGEMENTS

The authors would to thank the technical staff from the science workshops of Massey University, Victoria University of Wellington and Aachen University of Technology. This project was made possible by funding provided by the Foundation for Research Science and Technology.

6. REFERENCES

[1] Callaghan, P.T., "Principles of Nuclear Magnetic Resonance Microscopy", Oxford University Press, (1991).

[2] Eccles, C.D., Clark, C.J., Codd, S.L. and Dykstra, R., "Construction of an MRI system for Horticultural and Materials Science Research", Proceedings of the Fifth Electronics New Zealand Conference, 1998, pp 45-50.

[3] Hills, B.P. and Clark, C.J., "Quality Assessment of Horticultural Products by NMR", Ann Report NMR Spect, 50, 2003, pp 76-117.

[4] Parrot, L.J., Magazine of concrete research, 1991, 43, No. 154, pp 45-52.

[5] Bruker BioSpin GmbH, Silberstreifen 4, 76287 Rheinstetten, www.bruker-biospin.de/NMR.

[6] Eidmann, G., Savelsberg, R., Blumler, P. and Blumich B., "The NMR MOUSE, a Mobile Universal Surface Explorer", J. Magn. Reson, A 122, (1996), pp 104-109.

[7] NMR Apparatus. New Zealand patent application 520114 / International Patent Application No. PCT/ NZ03/00149

[8] Dykstra, R., "A DSP/USB platform for instrumentation and control", Proceedings of the Ninth Electronics New Zealand Conference, 2002, pp 6-11.

[9] Analog Devices, Corporate Headquarters, One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, www.analog.com.

Appendix D

System specifications

System core:
Interface: Full speed USB1.1
Pulse program timing resolution: 10 ns
Waveform memory: 32k, 24-bit samples
User Interface: Prospa
I/O: 24 TTL outputs

Receiver:
Input impedance: 50 Ω
Receiver input sensitivity: 1.1 Vpk-pk
Sample rate: 62.5 MHz
Resolution: 14-bit
Input bandwidth: 100 kHz to 200 MHz.
Receiver output maximum bandwidth: 2 MHz sample rate for both (I /Q) channels.
Filtering: User programmable digital.

Transmitter:
DDS output level: 0 dbm
Output impedance: 50 Ω
Clock rate: 125 MHz
Modulation capabilities: Frequency, phase and amplitude modulation.
Maximum update rate: 1 MHz

Receiver amplifier:
Gain: -20 to +70 dB, digitally controlled in 3 dB steps.
Input/output impedance: 50 Ω
Bandwidth: 1 kHz to 600 MHz

Power supply:
Lapspec and Modspec both use 24 V DC. The power demand is dependant on the RF power requirements.

Dimensions:
Lapspec: Custom 360 x 240 x 55 mm, 3.6 kg.
Modspec: Schroff, 3 U, 160 mm card depth, ¾ 19 inch width.