FPGA-based laser stabilisation using modulation transfer spectroscopy
Abstract

This project investigated the closed-loop frequency stabilisation of a diode laser using a heterodyne scheme known as modulation transfer spectroscopy. FPGA-based digital signal processing was used in place of a standard analogue approach. By digitally synthesising the driving signal for an acousto-optic modulator, unprecedented control over the modulation transfer spectroscopy error signal was achieved. Digital mixing and loop filtering provided performance comparable to an analogue approach. A control bandwidth of 150 kHz was achieved, limited by the diode laser controller rather than the digital system.
Acknowledgements

I would like to deeply thank my two supervisors, Assoc. Prof. Lindsay Kleeman and Dr. Lincoln Turner, without whose insight, inspiration and valuable experience this project would not have succeeded. Each has invaluably shaped my understanding and design philosophy, and I have benefited immensely from the complementary strengths of the science and engineering communities.

Thanks also to the other members of Dr. Turner’s research group, in particular Dr. Russell Anderson, Martijn Jasperse and Alex Wood. Without your assistance and encouragement, several parts of this project would have been left incomplete!

Finally, I would like to thank my family and friends, whose support and curiosity made this project all the more enjoyable. Thanks in particular to Georgia Rubira, who helped me improve this document at (literally) the last minute.
1 Introduction

The School of Physics, Monash University is organising a research laboratory to study Bose-Einstein condensates (BECs), headed by Dr. Lincoln Turner. A BEC is a metastable phase of matter that forms when certain dilute gases, such as rubidium-87, are cooled to very near absolute zero [1]. BECs have many unique and fascinating properties, beyond the scope of this document.

BECs were predicted by Bose and Einstein in the 1920s, but have only been experimentally realised in the last 15 years, thanks largely to the development of laser cooling and trapping techniques [1]. These allow the stringent temperature and density requirements of Bose-Einstein condensation to be met. In Dr. Turner’s laboratory, laser light will be used for cooling, trapping, imaging and more advanced manipulation of rubidium atoms, to create and investigate BECs.

This project is concerned with the frequency stabilisation of a diode laser, which will be the main source of laser light for the laboratory. The stabilisation was carried out using closed-loop control, and the technique of modulation transfer spectroscopy was used to generate an error signal for the controller. The novelty of this project lies in the use of a purely digital system to generate and optimise the error signal and the control loop.

I have adopted a slightly unorthodox layout in this chapter: the first sections present the background theory behind the project, then the motivations and requirements of the project are discussed. This order has been chosen to facilitate understanding.

Next we briefly discuss laser cooling, a critical role of the laser in the future, and illustrate why frequency stability is desirable.

1.1 Laser cooling

The type of laser cooling that will be used relies on the Doppler effect. We work with rubidium atoms, which have many atomic energy levels. We are concerned with two levels in particular, which between them form an atomic transition with energy \(E = h\nu_0\); \(h\) is Planck’s constant and \(\nu_0\) is the transition frequency. If we direct a photon of frequency \(\nu_0\) at a stationary rubidium atom, some of the time the atom will absorb the momentum and energy of the photon and begin to travel. It will eventually re-emit a photon in a random direction, receiving an equal and opposite momentum ‘kick’ (like the recoil of a cannon after firing a cannonball).

If the atom is initially moving towards the photon, the photon has a smaller chance of being absorbed, due to the Doppler effect shifting the photon frequency to \(\nu_0 + \nu_{doppler}\) in the atom reference frame. Lowering the original photon frequency to \(\nu_0 - \nu_{doppler}\) compensates for this - now an atom moving towards the photon source is likelier to absorb the photon than a stationary atom.
CHAPTER 1. INTRODUCTION

Figure 1.1: Effect of linewidth on cooling efficiency. For broad linewidths, a large fraction of photons do not cool the atoms, potentially warming them instead. The spectral shape is related to the spectrum of the laser frequency noise; white noise leads to Gaussian spectra while $1/f$ noise creates Lorentzian spectra [3].

Thus, moving atoms are likelier than stationary atoms to receive a momentum impulse from photons. Over time in a cloud of atoms, the average velocity in the face of incoming photons is reduced through this process. If we point six beams at the cloud from the $\pm x$, $\pm y$ and $\pm z$ directions, all at frequency $\nu_0 - \nu_{doppler}$, we can reduce velocity in all six axes — cooling the atom cloud!

For atoms at a temperature of 100 µK, the required photon detuning is $\nu_{doppler} \sim 300$ kHz. If we treat the photon frequencies as normally distributed with a mean $\mu = \nu_0 - \nu_{doppler}$, their standard deviation must be below $\sim 1$ MHz. Much larger, and the excess quantity of photons at $\nu_0 - \nu_{doppler}$ over photons at $\nu_0 + \nu_{doppler}$ would become only a fraction of the total photon quantity, as shown in Figure 1.1, and the cooling process would lose efficiency [2]. The linewidth of a laser is related to this standard deviation measure. We seek to both stabilise the laser frequency¹ (keep the mean frequency constant) and narrow its linewidth (minimise the standard deviation).

1.2 Laser locking

For rubidium, $\nu_0 = 384$ THz (780 nm infrared); thus a 1 MHz linewidth requires a stability of roughly 1 part in $10^8$. A laser particularly suited to our purpose (and budget) is the common laser diode, also used in CD burners. To achieve the required frequency stability, a system is built around the diode to control its temperature and current, and an external optical cavity is used to amplify the desired output modes [4]. This project relies on closed-loop control of the cavity length and diode current to stabilise the frequency and decrease

---

¹By passing it through an acousto-optic modulator, discussed in §2.2, the laser beam can be altered in frequency by a precise amount. Thus the mean frequency does not have to be $\nu_0 - \nu_{doppler}$, as long as it is within several hundred megahertz.
1.3. FREQUENCY MODULATION

the linewidth of the external-cavity diode laser (ECDL); closed-loop frequency control of a laser is known as laser locking.

The frequency of an ECDL is subject both to slow drifts, caused by effects such as temperature and pressure changes, and rapid jitter, due primarily to current and acoustic noise. Both of these can be reduced by a suitable controller; the design and construction of the controller is the subject of this project. Closed-loop laser frequency stabilisation is commonly called laser locking.

Error signal

A closed-loop controller requires information regarding its control variable; for example an automotive cruise control uses the speedometer to determine the difference between the desired and actual speed of a vehicle. Similarly a laser frequency controller requires an error signal which is proportional to the detuning of the laser from a frequency set point.

Our desired set point is the spectroscopic transition in rubidium-87 with frequency $\nu_0$, that will be used for cooling rubidium gas\(^2\). Creating this is a challenging problem, due to the very high accuracy required and the drift of most ordinary frequency sources. One candidate method is modulation transfer spectroscopy (MTS), which relies on a sample of rubidium vapour to generate a frequency reference. MTS produces an error signal as shown in Figure 1.2(b), and I investigated MTS in detail last year as part of my physics course [6, §1.2]. The details of the physics behind MTS are outside the scope of this thesis, but in the next sections I describe a basic mathematical model of the error signal, allowing an empirical understanding of MTS crucial for carrying out the work of this project.

1.3 Frequency modulation

MTS relies heavily on frequency modulated electromagnetic waves, at both electronic ($\sim 10^7$ Hz) and optical ($\sim 10^{14}$ Hz) frequencies. At a fixed point in space, the electric field amplitude $E_1(t)$ of an electromagnetic wave is described by

$$E_1(t) = E_0 \sin (2\pi f_c t), \quad (1.1)$$

where $E_0$ is the peak amplitude and $f_c$ is the carrier frequency. If the phase of the wave is modulated sinusoidally at a modulation frequency $f_m$, we obtain a frequency-modulated (FM) wave $E_2(t)$,

$$E_2(t) = E_0 \sin [2\pi f_c t + \delta \sin(2\pi f_m t)], \quad (1.2)$$

where $\delta$ is the modulation index of the FM signal. Note that the instantaneous phase of $E_2(t)$ is

$$\phi_i = 2\pi f_c t + \delta \sin(2\pi f_m t) \quad (1.3)$$

\(^2\)The actual frequency is $\nu_0 = 384.228115$ THz, or 780.246021 nm in vacuum [5]
and because frequency is the derivative of phase, phase modulation is equivalent to frequency modulation. The time derivative of \((1.3)\) divided by \(2\pi\) is the instantaneous frequency \(f_i\),

\[
f_i = \frac{d\phi_i}{dt} = f_c + \delta f_m \cos(2\pi f_m t)
\]  

(1.4)

which deviates by an amount \(\delta f_m\) from the carrier frequency \(f_c\). We define the frequency deviation \(f_D\) as

\[
f_D = \delta f_m.
\]  

(1.5)

In the frequency domain, \(E_2(t)\) is equivalent to a Fourier frequency \(f_c\) with sidebands spaced at intervals of \(f_m\), whose amplitudes are defined by \(J_n(\delta)\), the Bessel functions [7]:

\[
E_2(t) = E_0 \sum_{n=-\infty}^{+\infty} J_n(\delta) \sin \{2\pi [f_c + n f_m] t\},
\]  

(1.6)

Figure 1.3 shows \((1.4)\) and \((1.6)\) pictorially.

If the modulation index \(\delta\) is less than 1, \(J_n(\delta) \approx 0\) for \(|n| > 1\), and we can neglect all sidebands for which \(|n| > 1\). Now, \((1.6)\) simplifies to three terms:

\[
E_2(t) = E_0 \left\{ \frac{\delta}{2} \sin [2\pi(f_c + f_m)t] + \sin(2\pi f_c t) - \frac{\delta}{2} \sin [2\pi(f_c - f_m)t] \right\}.
\]  

(1.7)

Note that at \(t = 0\), the two sidebands are opposite in sign, thus phase-shifted by \(\pi\) from one another; any other phase shift introduces a component of amplitude modulation.
Figure 1.3: Simulation of a) the instantaneous frequency of an FM wave, given by (1.4), and b) the power spectrum of the FM wave, obtained using an FFT of the time-domain FM signal given by (1.2). The spectral peak amplitudes agree closely with (1.6). Parameters are $f_c = 20$ MHz, $f_m = 3$ MHz, $f_D = 2$ MHz. This leads to $\delta = 0.666$ and $J_1(\delta)/J_0(\delta) = 0.3494$; thus the first-order sidebands are -9.1 dB below the carrier (dBc). The second-order pair are -25 dBc and can be neglected.

1.4 Modulation transfer spectroscopy

In the following discussion, we refer to optical frequencies using $\nu$, and electronic frequencies using $f$. Let us label the frequency of the laser as $\nu_c$, and the desired laser frequency as $\nu_0$, which is the frequency of the cooling transition discussed in §1.1.

The MTS system involves first splitting the laser output into two laser beams: the pump and the probe beam. The pump beam is frequency modulated at frequency $f_m$ and modulation index $\delta$, while the probe beam is unmodulated. The pump beam electric field resembles (1.6):

$$E_{\text{pump}}(t) = E_0 \sum_{n=-\infty}^{+\infty} J_n(\delta) \sin \left\{ 2\pi [\nu_c + n f_m] t \right\}.$$  

The two beams are sent in opposite directions through a vapour cell containing rubidium gas, and are carefully overlapped. Through nonlinear four-wave mixing, beyond the scope of this document [6, §2.2], modulation from the pump beam is transferred to the probe beam. The probe beam is focused on a photodetector, where the sidebands of the beam beat with the carrier and one another [8]. If the probe beam is frequency-modulated, pairs of $n^{\text{th}}$-order sidebands are equal in amplitude and opposite in phase; the beats will hence cancel out and there will be no detected signal. Any component of amplitude modulation causes an unevenness in the sideband amplitudes and phases, producing a sinusoidal signal at the modulation frequency, whose amplitude and phase are dependent
on the detuning frequency, \( \nu_c - \nu_0 \);

\[
S(\Delta) = \frac{C}{\sqrt{\gamma^2 + f_m^2}} \sum_{n=-\infty}^{\infty} J_n(\delta) J_{n-1}(\delta) \left\{ (L_{(n+1)/2} + L_{(n-2)/2}) \cos(2\pi f_m t) ight. \\
+ \left. (D_{(n+1)/2} - D_{(n-2)/2}) \sin(2\pi f_m t) \right\} \quad (1.9)
\]

where

\[
L_n = \frac{\gamma^2}{\gamma^2 + (\Delta - nf_m)^2} \quad \text{and} \quad D_n = \frac{\gamma(\Delta - nf_m)}{\gamma^2 + (\Delta - nf_m)^2}; \quad \Delta = \nu_c - \nu_0. \quad (1.10)
\]

The \( L_n \) terms are known as *absorptive Lorentzian* functions, while the \( D_n \) terms are *dispersive Lorentzians*. Here \( \gamma \) is the natural linewidth of the \( \nu_0 \) transition (6.066 MHz for \(^{87}\text{Rb}\)), \( \Delta \) is the laser detuning from the atomic transition, and \( C \) is a function of the beam field amplitudes and physical constants. If we again assume \( \delta < 1 \), (1.9) simplifies to

\[
S(\Delta) = \frac{C}{2\sqrt{\gamma^2 + f_m^2}} \left\{ (L_1 + L_{-1/2} - L_{1/2} - L_{-1}) \cos(2\pi f_m t) + (D_1 - D_{-1/2} - D_{1/2} + D_{-1}) \sin(2\pi f_m t) \right\} \quad (1.11)
\]

The \( \cos(2\pi f_m t) \) term is known as the *in-phase* term, and \( \sin(2\pi f_m t) \) as the *quadrature* term. If we electronically multiply \( S(\Delta) \) by a signal \( M = \cos(2\pi f_m t + \phi) \) then low-pass filter the result (this process is known as *demodulation*; discussed in Ch. 5), we can recover either the in-phase or quadrature terms of (1.11) by simply altering the phase shift \( \phi \). A simulation of these equations for various modulation frequencies is shown in Figure 1.4 [6, §2.3].

Thus, MTS produces an error signal that crosses zero at \( \nu_0 \) and is suitable for laser locking\(^3\). By adjusting \( \phi \) and thus altering the ratio between the in-phase and quadrature components, we can optimise for signal amplitude, breadth or slope.

\(^3\)The optical scheme used actually causes the zero crossing to occur at \( \nu_0 + 80 \text{ MHz} \), due to the acousto-optic modulator applying an offset of 160 MHz to one of the beams. See §2.2 for details.
Maximising the signal-to-noise ratio

There are many techniques of generating an error signal that do not require frequency modulation of the laser beam [9–11], and as a result require simpler optics than MTS. MTS is worth the extra effort for two reasons. The first is that an MTS error signal crosses zero only at the frequencies where we wish to lock a laser, unlike most other techniques; if the laser suffers a disturbance it is very easy to re-centre on the desired frequency.

The second is that modulated (also called heterodyne) techniques such as MTS encode their error signal information at the modulation frequency, rather than baseband. Diode lasers suffer from amplitude and phase (frequency) noise, which can both be modelled as white noise added to $1/f$ noise [3]. At low frequencies the $1/f$ noise dominates, while above several hundred kilohertz the opposite is true and the amplitude and frequency noise floors are lower. Since MTS relies on detecting amplitude modulation at and around the modulation frequency $f_m$ (see previous section), amplitude noise near $f_m$ translates into voltage noise on the error signal, worsening the laser frequency noise as the controller tries to compensate for the phantom noise.

This problem can be reduced by maximising the amplitude of the MTS signal on the probe beam; simply increasing the system gain will not improve the SNR. This is discussed in Ch. 7.

1.5 Project outline

We have discussed the key background theory behind this project. The main part of this project consisted of generating and evaluating an MTS error signal using purely digital means, specifically the Ettus Research USRP2 software-defined radio peripheral. The remainder involved closed-loop control using this signal.

Basic requirements

The project hardware and software tasks were be broadly divided into two categories: digital implementation of the MTS error signal, and digital implementation of the laser controller.

The first goal required generation of an FM signal, which is discussed in Ch. 4, and demodulation/filtering of a photodetector signal, discussed in Ch. 5. These tasks took most of the time in this project.

Implementing a low-bandwidth laser controller was less difficult. A commercial diode laser controller (MOGlabs DLC-202) was used to run the laser both in open-loop mode for system diagnosis and optimisation, and closed-loop mode for frequency stabilisation. In the latter mode it relied on the error signal produced by the digital hardware. The control bandwidth is limited to $\sim 200$ kHz, however, which is less than the megahertz or more we desired (see requirements analysis for details). To attempt this, the error signal was fed directly into the laser diode, bypassing the controller bandwidth limit.

With these requirements in mind, we discuss the chosen implementation strategy.


**Implementation**

For this project, it was realised that instead of using analogue electronics, the signal processing could be carried out digitally using an Ettus Research USRP2 software-defined radio peripheral. This has a Xilinx Spartan 3 FPGA, connected to high-bandwidth ADCs and DACs suitable for both outputting 80 MHz FM to the AOM and receiving the photodetector signal from the MTS setup.

The next two chapters in this document lay the groundwork for a discussion of the HDL development later. Ch. 2 discusses the optics required to implement the MTS error signal, and Ch. 3 explains the electronic system architecture and design strategy.

**Note on thesis structure**

It has been a challenge writing an engineering thesis while incorporating enough background theory to allow a non-specialist to understand; additionally this document will be used as a laboratory reference by whoever needs to modify/optimise my system in the future. I have included more detail than a specialist reader requires, so please feel free to skim Ch. 2 and Ch. 3.
2 Optical system

While more development went into the electronic design, the optical layout and optimisation in this project were critical to its success. In this chapter the layout of the spectrometer is presented, and the properties of the components used in the MTS spectrometer are discussed.

The spectrometer relies on linearly and circularly polarised light, and this chapter assumes a basic understanding of the properties of half- and quarter-wave plates and polarising beamsplitters. For a brief discussion, please see [12].

2.1 Optical layout of MTS spectrometer

Figure 2.1 shows the key optical elements in the spectrometer. Currently, 20 mW of light enters the spectrometer from the laser; the remainder is sent to experiments requiring laser light in the laboratory. The light power will be reduced in the final system to around 5 mW, leaving more for the experiments.

The 20 mW of light is split into two beams by polarising beamsplitter PBS2, and the power ratio between them is adjusted by rotating the half-wave plate HWP1. The beam that passes through PBS2 is directed into a Galilean telescope consisting of L5 and L6, where it is narrowed by a factor of 6.5, resulting in a \(1/e^2\) diameter of \(\sim 100\ \mu\text{m} \).

This passes through the acousto-optic modulator, discussed below, and returns back on itself to polarising beamsplitter PBS1. Its polarisation has been shifted by 90° by passing twice through the quarter-wave plate QWP1, and it is now reflected by PBS1 towards PBS3. Having been rotated 90° by HWP2 it passes through PBS3 into a Galilean telescope consisting of L3 and L4, where it is expanded to a \(1/e^2\) diameter of \(\sim 2.5\ \text{mm} \). It propagates through the rubidium vapour cell, acting as the pump beam.

Meanwhile, the portion of the original beam reflected by PBS2 is expanded by L1 and L2 to the same diameter as the pump beam, and propagates through the vapour cell in the opposite direction, acting as the probe beam. It is reduced back to its original diameter by L3 and L4, and reflected to the photodetector PD1 by PBS3.

2.2 Acousto-optic modulator

An acousto-optic modulator (AOM) consists of a small piezoelectric actuator connected to a transparent crystal of a material such as tellurium oxide (TeO\(_2\)). If an RF signal is fed into the actuator, it vibrates at the RF frequency, inducing travelling waves of pressure in the transparent crystal. This is analogous to a vibrating diaphragm of an audio speaker inducing pressure (sound) waves in air. These waves cause the density and the refractive
Figure 2.1: Optical layout of the MTS spectrometer. PBS: polarising beamsplitting cube, HWP: half-wave plate, QWP: quarter-wave plate, L: lens, U: analogue amplifier, F: analogue filter, AOM: acousto-optic modulator, PD: photodetector. A fraction of the total laser light is directed into the spectrometer, where it is split into two beams by PBS2. One is frequency modulated using a double-pass AOM arrangement, and sent through the rubidium vapour cell as the pump beam. The other is sent in the opposite direction through the vapour cell as the probe beam, and its intensity is measured at PD1. Only key elements are shown; beam steering/focusing lenses/mirrors are omitted.

Optical frequency modulation theory

We can model the electric field amplitude of a light beam entering the AOM as

\[ E_{in}(t) = E_0 \sin \left( 2\pi \nu_c t \right), \]  

(2.1)

where \( E_0 \) is the maximum amplitude and \( \nu_c \) is the optical frequency (384 THz for our laser). Let \( \phi_i \) be the instantaneous phase of the RF signal entering the AOM.

Light entering the AOM is diffracted by the crystal, and altered both in phase (i.e. frequency) and direction as shown in Figure 2.2a. After this its electric field is described by

\[ E_{out}(t) = AE_0 \sin \left( 2\pi \nu_c t + \phi_i \right), \]  

(2.2)

where \( A \) is an attenuation factor \((A < 1)\) due to the imperfect diffraction efficiency of the AOM \( \text{[13]} \). From \( \text{[1.3]} \), the phase of an FM signal is

\[ \phi_i = 2\pi f_c t + \delta \sin(2\pi f_m t), \]
2.2. ACOUSTO-OPTIC MODULATOR

Figure 2.2: Acousto-optic modulator (AOM) being used for frequency modulation of the pump beam. 

**a)** Basic principle of AOM: a laser beam is diffracted by density waves induced in the crystal by an RF signal, altering its frequency and direction. 

**b)** When the RF is frequency-modulated, a ‘rainbow’ is produced by the AOM. We wish to remove the divergence and make the beam uniform. 

**c)** Double-pass arrangement used in this project. An unmodulated beam enters the AOM and is modulated twice, cancelling out its nonuniform frequency spread.

and thus, for an FM input signal, the AOM output beam is described by

$$E_{\text{out}}(t) = AE_0 \sin \left[ 2\pi (\nu_c + f_c) t + \delta \sin (2\pi f_m t) \right]. \quad (2.3)$$

Modulation of the RF frequency thus modulates the light frequency, but also its direction, producing a ‘rainbow’ as shown in [Figure 2.2b]. We wish to remove this effect, thus the beam must be reflected back into the AOM for a second pass. With suitable alignment \[14\], this cancels out the change in direction from the first pass. This ‘double-pass’ technique is used in the spectrometer to create frequency-modulated light as shown in [Figure 2.2c].

A double pass is equivalent to a double addition of the instantaneous phase to (2.1), which results in

$$E_{DP}(t) = BE_0 \sin \left[ 2\pi (\nu_c + 2f_c) t + 2\delta \sin (2\pi f_m t) \right], \quad (2.4)$$

where $B$ is the double-pass attenuation factor ($B \leq A^2$). This is an FM equation, and can be decomposed in the same way as (1.6) to obtain

$$E_{DP}(t) = BE_0 \sum_{n=-\infty}^{+\infty} J_n(2\delta) \sin \left[ 2\pi (\nu_c + 2f_c + nf_m) t \right], \quad (2.5)$$
and for $\delta < 0.5$ (analogous to $\delta < 1$ in (1.6)) we can neglect all but the first-order sidebands:

$$E_{DP}(t) = BE_0 \{ \delta \sin[2\pi(\nu_c + 2f_c + f_m)t] + \sin[2\pi(\nu_c + 2f_c)t] \\
- \delta \sin[2\pi(\nu_c + 2f_c - f_m)t] \}.$$ \hspace{1cm} (2.6)

The AOM is thus used to frequency modulate the pump beam.

**Driving the acousto-optic modulator**

There were two main requirements for the AOM to successfully operate: correct optical alignment and a suitable RF input. I followed the approach in [14] for optical alignment but my overall optical efficiency ($B^2$) was only 30%. This may have been because L5 and L6 had -15 and 100 mm focal lengths respectively, compared with -50 and 100 mm for the recommended setup. I used a shorter-focal-length concave lens to minimise the beam diameter entering the AOM, which increased the strength of modulation applied to the beam at high modulation frequencies (the modulation bandwidth) [15], however using a -25 or -30 mm lens may improve the optical efficiency of the system. This was not attempted during the project due to time constraints — re-alignment of the AOM apparatus takes several hours.

The AOM used has a rated frequency of 80 MHz, and a maximum RF power of 30 dBm. The modulation frequency $f_m$ was chosen to be 6.25 MHz. The USRP2 was used to produce the required FM; discussed in Ch. 4.

### 2.3 External-cavity diode laser and controller

The ECDL was built around a Sharp GH07895A6C laser diode at the University of Melbourne by Dr Robert Scholten’s research group [4], and loaned to our laboratory. Its external cavity length and diode current are controlled open-loop by a MOGlabs DLC-202 laser controller (referred to as the ‘MOGbox’ herein), which can also function as a closed-loop controller if it is provided with an error signal. This mode is discussed in §7.3.

The diode is designed for CD-burning applications with a room-temperature wavelength of 784 nm, and is cooled to reach the region of interest around 780 nm; this is accomplished by a Peltier cooler also controlled by the MOGbox. The frequency of a laser diode is also sensitive to the current supplied, and a diode current of between 75 and 83 mA was used, producing 40 mW of optical power.

The internal circuitry of the MOGbox has a bandwidth of 200 kHz, which was used to achieve control bandwidths of around 150 kHz. A current injection circuit was added into the system after a basic lock had been obtained; the details of the diode and the current injection circuitry are discussed in Ch. 8.

### 2.4 Photodetector

The photodetector is a Thorlabs PDA10A-EL silicon detector. Its role is twofold: to provide a DC saturated-absorption signal as shown in Figure 1.2a) that can be referred
to when manually tuning the laser frequency, and to generate an MTS signal at the modulation frequency \( f_m \), by the beating of the beam carrier frequency with the sidebands transferred inside the vapour cell. Figure 2.3 presents the roles of the AOM, vapour cell and photodetector.

Let us model the electric field amplitude of the probe beam at the photodetector as

\[
E_{PD}(t) = A \cos [2\pi \nu_c t] + B \cos [2\pi (\nu_c - f_m) t + \phi_B] + C \cos [2\pi (\nu_c + f_m) t + \phi_C],
\]

where \( B(\Delta) \), \( C(\Delta) \), \( \phi_B(\Delta) \) and \( \phi_C(\Delta) \) are the amplitudes and phase shifts of the lower and upper sidebands respectively; these terms vary as the laser detuning is varied (explicit dependence on \( \Delta \) is suppressed in the following discussion). Note that \( \nu_c \approx 380 \text{ THz} \) and \( f_m < 10 \text{ MHz} \).

The photodetector outputs a current proportional to the light intensity, which is pro-

Figure 2.3: Roles of the AOM, vapour cell and photodetector in modulation, modulation transfer and demodulation. Originally the pump and probe beams are unmodulated, as shown in a) and d). The double-passed AOM is driven with an FM signal b) from the USRP2, corresponding to (1.7), which applies frequency modulation to the pump beam c) corresponding to (2.6). This propagates through the vapour cell opposite to the unmodulated probe beam. Modulation is transferred from the pump to the probe beam, mediated by the rubidium vapour — the amplitude and phase of the probe sidebands is uneven, resulting in an amplitude and frequency-modulated beam e) corresponding to (2.7). The sidebands beat with the carrier at the photodetector to produce f), resulting in (2.9). This is used by the USRP2 to produce a control signal.
portional to the square of the electric field;
\[ I_{PD}(t) \propto \left\{ A \cos [2\pi \nu_c t] + B \cos [2\pi (\nu_c - f_m) t + \phi_B] + C \cos [2\pi (\nu_c + f_m) t + \phi_C] \right\}^2 \]

\[ = A^2 \cos^2 [2\pi \nu_c t] + B^2 \cos^2 [2\pi (\nu_c - f_m) t + \phi_B] + C^2 \cos^2 [2\pi (\nu_c + f_m) t + \phi_C] \]
\[ + AB \cos [2\pi \nu_c t] \cos [2\pi (\nu_c - f_m) t + \phi_B] \]
\[ + AC \cos [2\pi \nu_c t] \cos [2\pi (\nu_c + f_m) t + \phi_C] \]
\[ + BC \cos [2\pi (\nu_c - f_m) t + \phi_B] \cos [2\pi (\nu_c + f_m) t + \phi_C] . \tag{2.8} \]

Next we apply the identity \( \cos(x) \cos(y) = \frac{\cos(x + y) + \cos(x - y)}{2} \) to the six terms in (2.8), discarding all of the \( \cos(x + y)/2 \) terms since these would oscillate at terahertz frequencies, rendering them undetectable to the electronics. Thus we obtain

\[ I_{PD}(t) \propto \frac{A^2}{2} + \frac{B^2}{2} + \frac{C^2}{2} \]
\[ + \frac{AB}{2} \cos [2\pi f_m t - \phi_B] + \frac{AC}{2} \cos [2\pi f_m t + \phi_C] \]
\[ + \frac{BC}{2} \cos [4\pi f_m t - \phi_B + \phi_C] \tag{2.9} \]

The first three terms are DC, and are proportional to the total beam intensity. When the laser frequency is swept over the rubidium \( D_2 \) transitions, the beam intensity varies with the rubidium absorption, resulting in Figure 1.2(a).

The next two terms oscillate at \( f_m \), and convey MTS information. Their phasor sum is

\[ S(t) = \frac{A}{2} \sqrt{B^2 + C^2 + 2BC \cos(\phi_B + \phi_C)} \]
\[ \times \cos \left[ 2\pi f_m t + \arctan \frac{C \sin \phi_C - B \sin \phi_B}{C \cos \phi_C + B \cos \phi_B} \right]. \tag{2.10} \]

The amplitude of the final term is proportional to \( B \times C \), while the previous pair have \( A \) as a factor. Because \( A \gg B \) by 40 dB or more for MTS, this term is neglected for the remainder of this document.\(^1\)

\( B, C, \phi_B \) and \( \phi_C \) all vary with the detuning frequency \( \Delta = \nu_c - \nu_0 \). When \( \Delta = 0 \), the sidebands are affected equally by the rubidium vapour and the beam is purely frequency-modulated; i.e. \( B(\Delta) = C(\Delta) \) and \( \phi_B(\Delta) + \phi_C(\Delta) = \pi \). Now,

\[ \sqrt{B^2 + C^2 + 2BC \cos(\phi_B + \phi_C)} = B - C = 0, \tag{2.11} \]
resulting in zero signal amplitude; this occurs in the MTS spectrum at the zero crossing frequencies.

Applying further trigonometric identities and some atomic physics beyond the scope of this document \[18\], (2.10) becomes the MTS equation from the introduction:

\[ S(\Delta, t) = \frac{C}{2\sqrt{\gamma^2 + f_m^2}} \left\{ (L_1 + L_{-1/2} - L_{1/2} - L_{-1}) \cos(2\pi f_m t) \right. \]
\[ + \left. (D_1 - D_{1/2} - D_{-1/2} + D_{-1}) \sin(2\pi f_m t) \right\} , \tag{1.11} \]

\(^1\)It is only present when the laser detuning \( \Delta \) is small, and could be used as a ‘lock detector’.
where all terms are defined in §1.4. Note that this assumes (2.7) is pure FM, i.e. of the same form as (1.7), which is only true if

\[
B = C \quad \text{and} \quad \phi_B + \phi_C = \pi
\]  

in the pump beam. Any distortion in the pump beam introduces residual amplitude modulation (RAM). §7.1 discusses the novel approach to minimising RAM developed in this project.

This concludes the discussion of the optics used in the experiment. The next chapter presents the overall control system, and the role of the optical setup therein.
3 Electronic system overview

This chapter presents an overview of the electronics used in this project, chiefly the Ettus Research USRP2 \[19\]. This was the core of the project, and the initial month was spent on testing its components and planning the system.

3.1 Basic requirements

The main goal of the system is producing a control signal, which is then used to drive actuators and alter the laser frequency. This is accomplished with minimal latency, as delays reduce the system control bandwidth.

The optics required for MTS have been discussed in the previous chapter. As discussed in §2.2, the AOM is driven by an FM signal with carrier and modulation frequencies $f_c = 80$ MHz and $f_m \geq 3$ MHz, whose power is $\sim 30$ dBm. The higher the modulation frequency the larger the SNR of the error signal, assuming the amplitude of the error signal is constant; this assumption is discussed in Ch. \[7\]. Various modulation frequencies were tested; $6.25$ MHz was selected for reasons discussed in §7.1.

We have seen in §2.4 that a sinusoid with frequency $f_m$ is produced by carrier-sideband beating in the probe beam. The equation is described by (1.9). To obtain a DC-coupled error signal this sinusoid must be demodulated, by multiplying it by a sine from a digital local oscillator with an adjustable phase and lowpass filtering the result. This process is carried out with minimal latency, as unlike the FM generator it is part of the control loop. It is discussed in Ch. \[5\].

The USRP2 is also used to modify the transfer function of the controller, by filtering the error signal and altering its gain and phase. This is discussed in §8.

3.2 System architecture

The relationship between the optical setup, USRP2, current injection circuit and MOGbox is shown in Figure 3.1. Figure 3.2 shows the key USRP2 systems.

3.3 HDL design

Motivation for redesign

The Ettus Research USRP2 is designed for software-defined radio, in which it is used as a transceiver while a PC carries out most of the signal processing and other operations,
transferring data to/from the USRP2 over gigabit ethernet. The FPGA design files produced by Ettus Research are open-source, and I initially hoped that they would require only minor modification. Documentation is very sparse, however, and it was difficult to identify what elements required reconfiguration. Additionally, the USRP2 firmware available at the start of this project was only compatible with GNU/Linux (due to the use of native Ethernet, rather than higher-level protocols such as TCP/UDP), I decided to create the system anew and avoid potential difficulties associated with an unfamiliar operating system and reverse-engineering a sparsely documented system.

Fortunately, the USRP2 FPGA is readily reprogrammable using a Xilinx USB programmer, which interfaces to the JTAG pins of the FPGA. This also provided the benefit of Chipscope as well as run-time CPU debugging, discussed in Ch. 6.

Software

HDL for the FPGA was written in the Xilinx ISE package (analogous to Altera Quartus). The modules in the design were first tested and optimised as individual systems, then combined together in a larger project for the final implementation. Simulation was carried out mainly in Icarus Verilog using testbenches, which had the advantage of zero compilation time and an excellent waveform viewer, GTKwave. Icarus Verilog was used extensively in debugging the IIR filters, discussed in §5.3.

Several elements, such as the SPI interface, FM generator and demodulation multiplier, required run-time debugging. This was carried out in Xilinx Chipscope (analogous to Altera SignalTap). Before the soft-core CPU had been implemented, Chipscope was also used as control interface; control lines within the FPGA could be altered through the Chipscope software in real-time.
3.4 Initial testing of the USRP2

Rewriting the FPGA bitstream proved more challenging than imagined. Early tests included flashing the LEDs, outputting a digitally generated sine wave from the DAC, and importantly, measuring the latency of the USRP2 from the ADC input to the DAC output.

Latency of the USRP2

This was measured by creating a bitstream that read the ADC data, clocked it through a register, and output it to the DAC. By inputting a square wave into the ADC, the minimum latency of the USRP2 was identified based on the delay between rising edges, shown in Figure 3.3.

The latency of the ADC and FPGA were 90 ns, roughly as expected; the ADC has a five-stage pipeline (50 ns) and the FPGA delays caused the remainder. Cable delays were calculated to be 3 ns. The major source of latency was the DAC, with its internal filters...
3.4. INITIAL TESTING OF THE USRP2

The total latency is 350 ns; 90 ns due to the ADC and FPGA, and 260 ns due to the DAC. This causes a $\pi$ phase shift in an incoming sinusoid at a frequency of 1.4 MHz; this is the best-case control bandwidth. A more precise test of latency is carried out in §8.3.

causing 260 ns or 26 clock cycles of delay. There appears to be no way to avoid this; it was realised that the DAC was poorly suited to closed-loop control!

Clocking

During the first two months of this project, a series of unusual issues was observed; seemingly-random changes in clock frequency, glitches on the ADC or DAC outputs, and occasional failure of the JTAG connection to the FPGA. This was eventually traced to the bitstream, which treated the input clock as single-ended while it was differential in reality. Additionally, in the original Ettus Research bitstream the clock was input into a PLL (presumably to reduce phase noise), whose output was then distributed across the FPGA — rather than making these changes piecemeal in my code, I incorporated the Ettus Research top-level module into my project and used it as a ‘wrapper’ for my original top-level module. After these steps were taken, the mysterious issues vanished.

There were several other problems that caused delays, both hardware and software, but clocking was the most major. Once this had been resolved, development of the FM generator was begun. This is discussed in the next chapter.
4 Frequency-modulated signal generation

As mentioned in §2.2, the AOM used in the spectrometer must be driven by an FM signal, with its carrier frequency $f_c$ around 80 MHz and an amplitude of $\sim 30$ dBm. This chapter discusses how this was accomplished using digital signal generation.

4.1 Signal generation on the FPGA

Motivation

The standard analogue approach to driving acousto-optic modulators is the voltage-controlled oscillator (VCO), such as the Minicircuits ZX95-78-S+ [20]. This outputs a sinusoid at a frequency that is nominally proportional to a control voltage. If this voltage is modulated sinusoidally, the VCO outputs an FM signal similar to (1.2). A VCO has several disadvantages for experimental work: its frequency and amplitude vary with temperature and age, it suffers from nonlinearities, and when generating FM its modulation bandwidth is limited. The VCO can be locked to a crystal oscillator using a phase-locked loop (PLL) circuit, minimising frequency drift, but its other drawbacks remain.

These can be avoided by the technique used in this project: direct digital synthesis of the FM waveform. Within the USRP2, a 16-bit 2’s complement signal is generated in the FPGA and output to the AD9777 digital-to-analogue converter (DAC). The frequency and amplitude drifts are now directly determined by the drift in the USRP2 clock oscillator frequency and DAC reference amplitude, discussed in §4.2. A digital approach offers great flexibility, allowing any combination of carrier, modulation and deviation frequencies — this is unlike virtually any analogue approach except a dedicated FM-capable function generator!

Generation strategy

As a basic DSP building block I used the Xilinx DDS IP core. A DDS consists of a phase word and a lookup table (LUT). On each clock cycle the phase is incremented by an amount proportional to the desired frequency, and the LUT outputs the corresponding amplitude at that phase. In this project it was more convenient to increment the phase word manually, with the LUT acting as a black box representing the $a \rightarrow \sin(a)$ function.

The next step was designing an FM synthesis strategy. The simplest version is setting up one DDS to output a sine wave at the modulation frequency:

$$\sin(2\pi f_m t),$$
4.1. SIGNAL GENERATION ON THE FPGA

multiplying this by a scale factor $\delta$ and adding it to a linearly incrementing phase to produce

$$\phi = 2\pi f_c t + \delta \sin(2\pi f_m t).$$

This is used as the input to a second DDS, to obtain

$$\sin \left[ 2\pi f_c t + \delta \sin(2\pi f_m t) \right],$$

which matches the basic FM equation, (1.2).

An alternative version is slightly more elaborate, and is based on

$$S_{FM}(t) = A \cos \left[ 2\pi \nu_c t \right] + B \cos \left[ 2\pi (\nu_c - f_m) t + \phi_B \right] + C \cos \left[ 2\pi (\nu_c + f_m) t + \phi_C \right] \quad (4.1)$$

which equals (1.7) if $A = 1$, $B = C = \delta/2$ and $\phi_B + \phi_C = \pi$. This can be generated by simply adding three sinusoids. The two alternatives are presented in Figure 4.1. The adjustable gains are implemented using multipliers with variable gain coefficients. The range of the 16-bit phase registers, $[0000\text{ }_{16}, \text{FFFF}_{16}]$, correspond to $[0, 2\pi]$; the $\pi$ phase relationship between the sidebands is synthesised by altering the initial sideband phases such that their sum is $8000\text{ }_{16}$.

The latter 3-sinusoid method was chosen mainly because the amplitudes and phases of the sidebands could be altered directly, which was advantageous for reasons discussed in the next sections. The slight overhead in resources was insignificant.

Digital-to-analogue conversion and daughterboard modification

We have discussed the required FM parameters in previous chapters: a carrier frequency of 80 MHz and a modulation frequency of $\geq 3$ MHz. Because the FPGA and DAC clock is only 100 MHz, the Nyquist sampling theorem only permits a maximum un-aliased output of 50 MHz. This was a major problem, and was eventually overcome by taking advantage of signal aliasing on the DAC output.

When the project began, the USRP2 was chosen partly because the AD9777 DAC was specified as being 400 MSPS with a clock rate of 100 MHz. The DAC achieves this by generating a 400 MHz clock using a PLL and interpolating between the 100 MSPS data samples to produce a 400 MSPS output. Noise and spurs are removed using digital filters, leaving a clear from 50 to 350 MHz. Other modes allow lower sampling frequencies: the mode which I had originally planned to use works by first interpolating to 200 MSPS. This produces the original 20 MHz signal with an image at 180 MHz, with digital filters attenuating the 50-150 MHz region. The result is then modulated at 100 MHz, shifting the 20 MHz signal to $(100 + 20 =) 120$ MHz, and the 180 MHz image to $(180 - 100 =) 80$ MHz$^1$. It was found that though this mode had the desired outcome, the modes of the DAC channels could not be altered independently; because the other channel could not produce a DC-coupled error signal with modulation, it could not be used.

Another strategy was attempted. When the digital input to the AD9777 DAC was at 20 MHz and interpolation was off, the DAC produced both the 20 MHz signal and an

$^1$Please see [21] for a very useful online simulator of the DAC and its properties.
Figure 4.1: Two alternative methods of digitally generating FM; \textbf{a)} phase-modulating a DDS or \textbf{b)} directly synthesising the carrier and sidebands and adding them together. By adjusting the relative phase of the sidebands in \textbf{b)}, FM or AM can be obtained. The phase initialisation logic is omitted.

alias at \((100 - 20 =) 80\,\text{MHz}\). The \(\sin(x)/x\) zero-order hold rolloff of the DAC attenuated the alias by \(\sin(80\pi/100)/(80\pi/100)\) or 12.6 dB [22], and the 30 MHz lowpass filter on the LFTX daughterboard reduced it further to roughly -40 dB below the 20 MHz signal. This filter was disabled by removing three capacitors on the LFTX as shown in Figure 4.2 allowing the alias to be used. This required altering the default DAC settings over SPI.

Additionally a second alias at 120 MHz was present, though weaker by 10 dB, which distorted the FM signal slightly. This was not removed, because it was well outside the AOM 3 dB bandwidth.

The strategy of using aliasing may be useful in the future if another type of AOM is used, such as a 180 MHz model. The DAC can be placed in 2x interpolation mode to create an alias at 180 MHz, with a similar approach to the above.
Figure 4.2: DAC output stage on LFTX daughterboard. The AD9777 DAC drives current through the path from IOUTP_B to IOUTN_B; the resultant voltages across R35/R36 are amplified by U2 and output at J49. The capacitors outlined in red were removed to allow the first DAC alias to be used in driving the AOM. Adapted from [23].

Analogue filtering and amplification

The 80 MHz alias was weak relative to the 20 MHz signal, which was solved by removing the 20 MHz signal using a Mini-circuits BHP-50+ 50 MHz highpass filter [24] and amplifying the signal to 30 dBm using a 24 dB Mini-circuits ZFL-500LN+ [25] followed by a Delta RF LA-2 power amplifier [26]. In Figure 2.1 these parts are F1, U1 and U2 respectively.

Control parameters

We have seen the means by which 80 MHz FM was produced. Table 4.1 lists typical configuration settings of the FM generator module when used to produce FM with $f_c = 80$ MHz and $f_m = 6.25$ MHz.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Notes</th>
<th>Bit width</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_c$</td>
<td>carrier frequency</td>
<td>16</td>
<td>3333$_{16}$ (20 MHz)</td>
</tr>
<tr>
<td>$f_m$</td>
<td>modulation frequency</td>
<td>16</td>
<td>1000$_{16}$ (6.25 MHz)</td>
</tr>
<tr>
<td>$A$</td>
<td>carrier amplitude</td>
<td>8</td>
<td>1B$_{16}$ (0.086)</td>
</tr>
<tr>
<td>$B$</td>
<td>lower sideband amplitude</td>
<td>8</td>
<td>23$_{16}$ (0.137)</td>
</tr>
<tr>
<td>$C$</td>
<td>upper sideband amplitude</td>
<td>8</td>
<td>0F$_{16}$ (0.059)</td>
</tr>
<tr>
<td>$\phi_B$</td>
<td>lower sideband phase offset</td>
<td>16</td>
<td>4800$_{16}$ (0.563\pi)</td>
</tr>
<tr>
<td>$\phi_C$</td>
<td>upper sideband phase offset</td>
<td>16</td>
<td>4B00$_{16}$ (0.586\pi)</td>
</tr>
</tbody>
</table>

Table 4.1: Typical settings used for FM generator

Values for $f_c$ and $f_m$ were calculated in MATLAB using dec2hex() and hex2dec() decimal-hexadecimal conversion, and did not require any modification. Values for $A$, $B$, $C$, $\phi_B$ and $\phi_C$ were initially calculated, then optimised using a spectrum analyser and an oscilloscope to observe the produced FM. These parameters are discussed in the next section.
CHAPTER 4. FREQUENCY-MODULATED SIGNAL GENERATION

Figure 4.3: Time-domain view of the FM generator output; \( f_c = 25 \text{ MHz} \), \( f_m = 780 \text{ kHz} \) and \( B/A = C/A = 0.1 \). a) The sidebands are in-phase, leading to pure AM — this can be inferred from the amplitude envelope oscillating at 780 kHz; b) the sidebands are out of phase, leading to FM, as seen from the minimal amplitude oscillation. A small amount of amplitude oscillation (\( \sim 3\% \)) remains, due to the lack of higher-order FM sidebands; this is inherent to the 3-sideband generation scheme, and is more significant for higher deviations.

4.2 Performance

Time domain

An oscilloscope was used to observe the output of the DAC, with screenshots shown in Figure 4.3. The carrier \( f_c \) is 25 MHz, with modulation frequency \( f_m = 780 \text{ kHz} \). Between the two images, only the offset phases \( \phi_B + \phi_C \) were altered: in a) this is 0, in b) this is \( \pi \). \( B \) and \( C \) are 0.1 times \( A \), thus a sideband-to-carrier ratio of 20 dB is shown.

Frequency domain

Figure 4.4 shows the output directly from the DAC. The three tones near 20 MHz are the original signals, with images at 80 and 120 MHz. The amplitudes are deliberately uneven at 20 MHz, to compensate for the differential attenuation caused by the \( \sin(x)/x \) envelope and analogue output stage. The end result is three tones of equal amplitude at 80 MHz. The unwanted 120 MHz images are weaker than the 80 MHz images by only 8 dB, which is undesirable; in practice the limited AOM bandwidth of 60-100 MHz attenuates them by 40-60 dB until they are insignificant.

Spurious-free dynamic range

Figure 4.4a) shows many spurs across the spectrum. When the BasicTX daughterboard was substituted instead of the LFTX, the spurs were not visible; this was because the BasicTX used transformers to couple the DAC to the output rather than op-amps. The op-amps of the LFTX cause weak intermodulation, which is responsible for the spurs. Because the second channel of the DAC was used for producing a DC error signal, the BasicTX could not be used.

As mentioned, the signal was sent through a 50 MHz highpass filter followed by 24 dB and 30 dB amplifiers before being used to drive the AOM. The filtering and 24 dB amplifi-
4.2. PERFORMANCE

Figure 4.4: a) Broadband scan of FM produced by the DAC; b) scan of the critical tones within the AOM passband. Spectrum analyser RBW = 30 kHz, VBW = 10 kHz. Spectra are attenuated by 10 dB.

The spectrum analyser view gave useful information about sideband amplitude and intermodulation, however an MTS error signal was required to optimise the amplitudes and phases more precisely. This is discussed in §7.1.

Drift

The oscillator frequency drift is $< 3$ ppm per year [27], while the DAC reference is a temperature-compensated bandgap, which is rated to 50 ppm per degree [28]. These minuscule values provide far more precision than is needed in this project. Additionally, any noticeable frequency or amplitude drifts can be cancelled by adjusting the carrier frequency and/or amplitude of the generated signal.

A qualitative experiment was carried out during the early stages of this project, where I digitally multiplied the input from the ADC by a local sinewave and sent the result to the DAC. By supplying a frequency into the ADC that was painstakingly matched to the internal USRP2 frequency, the output was a frequency-doubled sinusoid whose DC level remained constant. When I warmed up the USRP2 oscillator, the DC level would begin to vary sinusoidally due to the beat between the two sinewaves (see Ch. 5 for details), and after several minutes of warming the frequency was $\sim 30$ Hz. Thus, within an order of magnitude, the USRP2 crystal frequency altered by 30 Hz per 15°C (room temperature was 20°C, my finger was $\sim 35$°C), or 2 Hz/°C. Everyday temperature changes would be unlikely to affect the performance of the USRP2 significantly.
Drift due to ageing of the crystal was not established, but is likely more significant than temperature drift. The USRP2 circuitry allows an external clock reference \[29\], and if drift becomes a problem during the USRP2 lifetime, an external reference can be used instead of the on-board 10 MHz oscillator.

The three-sideband FM technique proved to be very flexible, and simultaneous amplitude and phase adjustment offers an extra degree of freedom beyond standard FM generation techniques, which is particularly suited to MTS. \[7.1\] examines this in detail.
Figure 4.6: Spectra of FM produced by the filter and two amplifiers for different FM amplitude settings. The FM amplitude for the light trace was weakened by 20 dB, with the output showing a 10 dB SFDR improvement and much fewer strong spurs. Power of weakened trace has been raised by 20 dB after data collection to facilitate comparison. Spectrum analyser RBW = 300 kHz, VBW = 100 kHz.
5 Demodulation

We have seen in §1.4 and §2.4 that the photodetector produces a signal at the modulation frequency \( f_m \). To produce a DC error signal, this must be demodulated; this chapter discusses how this was carried out using the FPGA.

5.1 Background theory

In §2.4 it was shown that the photodetector produces an electronic sinusoid of the form (1.11). For conciseness we write it as

\[
S(\Delta, t) = P(\Delta) \cos(2\pi f_m t + \phi) + Q(\Delta) \sin(2\pi f_m t) + N(t)
\]  

where \( N(t) \) is noise, predominantly 1/f laser noise mentioned at the end of §1.4. \( S(\Delta, t) \) is multiplied by \( \cos(2\pi f_m t + \phi) \), in a process known as mixing. Applying the identities \( \cos(x) \cos(y) = [\cos(x+y) + \cos(x-y)]/2 \) and \( \sin(\alpha) = \cos(\alpha - \pi/2) \), the result is

\[
S_M(\Delta, t) = P(\Delta) \cos(\phi) - Q(\Delta) \sin(\phi) + P(\Delta) \cos(4\pi f_m t + \phi) + Q(\Delta) \sin(4\pi f_m t + \phi) + N(t) \cos(2\pi f_m t + \phi)
\]  

The first pair of terms in (5.2) are time-independent, while the second pair oscillate with frequency \( 2f_m \). The noise term occupies a spectral region centred around \( f_m \), falling in power as \( 1/|f - f_m| \) around it (see Figure 5.1). The upconverted noise at \( f_m \) and the frequency-doubled pair at \( 2f_m \) are removed using a lowpass filter. The multiplication and filtering process is known as demodulation, and produces a signal of the form

\[
S_{DM}(\Delta) = P(\Delta) \cos(\phi) + Q(\Delta) \sin(\phi)
\]

This is the error signal, and its simulated dependence on \( \Delta \) is shown in Figure 1.4 for \( \phi = 0 \) and \( \pi/2 \). By altering \( \phi \) we may create a weighted sum of the in-phase \( P(\Delta) \) and quadrature \( Q(\Delta) \) components to optimise the error signal properties.

The digital multiplication was straightforward to implement; the filtering was more difficult. Figure 5.1 shows the completed demodulation system. Next we discuss its individual elements.

5.2 Variable-gain multiplier

This module consists of an internal phase register and sine lookup table similar to those used in FM generation, a variable gain to amplify the input signal from the ADC, and a multiplier whose output is (5.2).
5.3. DIGITAL FILTER DESIGN

The only important design in this section was writing a variable-gain stage with a wide dynamic range; this was accomplished by left-shifting the ADC signal by a variable amount. This was adjustable from 0 to 15. The with high gains caused quantisation noise to become significant; thus a 24 dB Mini-circuits ZFL-500LN+ amplifier was used (U3 in Figure 2.1). With the pre-amplifier present, shifts of between 3 and 5 were used (gains of 18 - 30 dB).

5.3 Digital filter design

We have seen that to demodulate a signal, multiplication is required to convert from the carrier frequency to baseband, and filtering is required to remove the frequency-doubled and 1/f noise components in the resultant signal.

Introduction

The following discussion assumes the reader is reasonably familiar with the concepts of the z transform, transfer functions, poles and zeros.

There are two major categories of digital filter: finite-impulse-response (FIR) and infinite-impulse-response (IIR). FIR filters are purely feed-forward, have linear phase response and are quite straightforward to design, while IIR filters incorporate feedback loops and are usually more challenging to design. Only zeros can be synthesised using an
FIR filter, while an IIR filter can synthesise both zeros and poles. Another important advantage of IIR filtering is lower latency, owing to the lower filter order generally required to achieve the desired performance. IIR filtering was chosen for this project.

The first step was choosing the filter parameters. It required two zeros at $f_m$ and $2f_m$ to minimise the frequency-doubled signal and $1/f$ noise, a passband up to $f_m$, a stopband attenuation of 20 dB or more, and reasonably small group delay. The filter design and analysis toolbox UI (FDAtool) in MATLAB was used to automatically calculate the coefficients for a fourth-order Cauer (elliptical) filter with the passband and stopband parameters mentioned above. A Cauer filter features zeros on the unit circle, and for a fourth-order filter, one complex conjugate pair is close to double the frequency of the other pair. This was taken advantage of, and the filter required little adjustment to attain zeros at $f_c$ and $2f_c$. A pole-zero plot and simulated transfer function are shown in Figure 5.2.

A common strategy in digital filter design is to break a high-order filter into second-order biquadratic (biquad) stages, whose transfer functions are of the form

$$\frac{Y(z)}{U(z)} = \frac{a + bz^{-1} + cz^{-2}}{1 + dz^{-1} + ez^{-2}}. \quad (5.3)$$

Each stage implements a pair of poles and zeros \[30\]. The stages can then be cascaded in series (output of one stage is input to the next) or parallel (outputs are added together simultaneously). Due to time constraints a series approach was used, though in a parallel approach would have saved a clock cycle of latency, though potentially sacrificing numerical accuracy or stability.

The coefficients for each biquad stage were automatically generated in MATLAB. Table 5.1 shows the coefficients used for different modulation frequencies and stopband attenuation.
### Coefficient 3.125 MHz 5.299 MHz 6.25 MHz

<table>
<thead>
<tr>
<th>Biquad stage 1</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>-1.8477</td>
<td>-1.5727</td>
<td>-1.4142</td>
</tr>
<tr>
<td>d</td>
<td>-1.8181</td>
<td>-1.8445</td>
<td>-1.7587</td>
</tr>
<tr>
<td>e</td>
<td>0.8372</td>
<td>0.8591</td>
<td>0.7943</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Biquad stage 2</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>-1.9616</td>
<td>-1.8901</td>
<td>-1.8478</td>
</tr>
<tr>
<td>d</td>
<td>-1.9494</td>
<td>-1.9153</td>
<td>-1.8639</td>
</tr>
<tr>
<td>e</td>
<td>0.9801</td>
<td>0.9574</td>
<td>0.9520</td>
</tr>
</tbody>
</table>

Table 5.1: Filter coefficients used in IIR biquad stages. \( a = 1 \) and \( c = 1 \) for all filters.

### Topology

As other digital signals, IIR filters may be fixed-point or floating-point. Though a floating-point filter usually has greater dynamic range and less numerical error than its fixed-point equivalent, it requires significantly more resources and has larger propagation delays (and is more difficult to design and debug!). For these reasons, a fixed-point filter was implemented.

**FDAtool** features automatic Verilog/VHDL exporting, with customisable options including internal multiplier sizes, accumulator lengths and fractional precision. Upon trying to compile a biquad in Xilinx ISE, however, the critical-path propagation delay was found to be \( 25 \text{ ns} - 15 \text{ ns} \) over the maximum delay permissible. Thus a fourth-order IIR filter was written manually, composed of two biquad blocks.

### HDL implementation

The Transposed Direct-Form I topology [30] was chosen for the filter, chiefly due to its ease of design but also due to its numerical stability. Its system is represented by a rearrangement of (5.3):

\[
Y(z) [1 + dz^{-1} + ez^{-2}] = U(z) [a + bz^{-1} + cz^{-2}]
\Rightarrow Y(z) = aU(z) + z^{-1} \left\{ bU(z) - dY(z) + z^{-1}[cU(z) - eY(z)] \right\}
\]

(5.4)

Converting to the time domain, we obtain the general biquad difference equation:

\[
y(k) = au(k) + bu(k-1) - dy(k-1) + cu(k-2) - ey(k-2)
\]

(5.5)

Note that in the special case of an elliptical filter, \( a = 1 \) and \( c = 1 \). This topology and critical paths are shown in Figure 5.3a; an extra register was added to reduce propagation delay as shown in Figure 5.3b). Once the coefficients of the two biquad stages had been selected, simulations were carried out in MATLAB with \texttt{fix()} being used to simulate fixed-point arithmetic (the fixed-point toolbox was unavailable). The results were encouraging, and virtually indistinguishable from Figure 5.2a).

Next the design was implemented in Verilog. Icarus Verilog and Xilinx ISIM were used to run a testbench, to allow functional simulation. A MATLAB script was used...
to generate a frequency chirp, which was input as the testbench stimulus; the testbench response was written to a text file and imported back into MATLAB. By comparing the MATLAB transfer function simulation to an FFT of the testbench results, the system was rapidly debugged.

The filter design went through several iterations until the internal bit lengths and structure performed comparably to the MATLAB simulation. The Verilog code was then compiled in Xilinx ISE, however the system violated setup time constraints by 1.5 ns due to the multiplier-adder-adder path shown in Figure 5.3. The filter coefficients were hard-coded, instead of being reprogrammable as had been hoped initially; an attempt to interface them to the controlling CPU resulted in compilation failure due to setup time violation. The FPGA layout was altered using Xilinx PlanAhead to allocate the filter modules to small areas of the FPGA, but I was unable to improve on the automatic router by more than 0.2 ns.

Fundamentally the filter is limited by the propagation delay through a multiplier and two adders. This is a limitation shared by all IIR filters, however pipelining techniques exist to avoid them at the cost of larger latency [31]. These were investigated briefly, but abandoned due to the increased latency (2-3 more clock cycles per biquad).

Another strategy in the future will be to use Xilinx MegaCores to implement optimised adders (such as carry-lookahead/carry-save), which may reduce this delay; this was not investigated due to time constraints. Despite the setup time violation, the filter stages functioned without any problems, likely due to conservative delay estimation within Xilinx ISE.

5.4 Performance

 Initially the filter was tested using a pulse generator as shown in Figure 5.1. This creates a series of pulses at the modulation frequency (the pulse frequency is unimportant; it was merely convenient to control both the pulse generator and demodulator using the same control bus). The Fourier transform of a series of pulses is a series of frequency-
domain spurs known as a frequency comb, which are spaced by the pulse frequency. The frequency comb was thus used as a run-time test of the filter performance: the impulse response was observed by viewing the filter output through Chipscope, while the amplitude envelope was observed using a spectrum analyser. Figure 5.4 demonstrates this. The filter performed well, although the attenuation at the 3.125 MHz notch was incomplete; nonetheless there was 40 dB attenuation at the notches, which was better than imagined. Next the demodulation system as a whole was tested by sending white noise into the ADC from a function generator. The output is shown in Figure 5.5. Because the energy was distributed over the entire spectrum the filter output is 12 dB weaker than Figure 5.4b), but performs similarly. The imperfect attenuation at the notches is due to the DAC noise floor rather than a flaw in the filter\textsuperscript{1}. Finally, the system was used to demodulate individual tones at 3 MHz; the results are shown in Figure 5.6. The result is a strong tone at 125 kHz, as expected, with weaker tones at 3.125 MHz and 6.125 MHz. Because the input frequency is 3 rather than 3.125 MHz, the notch at 6.25 MHz does not attenuate the signal as strongly as it would an input tone at 3.125 MHz. Altering the digital gain does not worsen SNR; the noise grows by a smaller factor than the tones. The above tests were carried out using a 3.125 MHz modulation frequency. Performance was not characterised in detail for 6.25 MHz.

\textsuperscript{1}Agreement was improved when white noise was added to the simulation output; this is omitted for brevity.
Figure 5.5: Demodulation of white noise, demonstrating the filter with an analogue input. Spectrum analyser RBW = 3 KHz, VBW = 300 Hz.

5.5 Future work: moving-average filter

Close to the project deadline I recalled that a moving-average FIR filter has zeros spaced at integer multiples of the inverse of its period. For instance, an 8-sample moving average nullifies any signal whose period is 8, 4 or 2 samples. It was thus realised that a 16-sample moving average would have zeros at 6.25 MHz and 12.5 MHz; additionally it would not require any hardware multipliers.

Figure 5.6: Demodulation spectra produced for a 3 MHz input at a) -15 dBm, and b) -50 dBm. The true MTS input was \( \sim -20 \) dBm. The bottom green line is the spectrum analyser noise floor; settings were RBW = 3 KHz, VBW = 300 Hz.
A MATLAB simulation was carried out, in which each input sample was stored inside
an array element whose index was given by a modulo-16 counter incremented every clock
cycle. The sample was also added to an accumulator, and the array element whose index
was one ahead of the current counter was subtracted from the accumulator. Thus the
accumulator contained the sum of the last 16 samples; this recursive scheme requires only
registers, multiplexers and adders in HDL\textsuperscript{2}.

While its ‘stopband’ attenuation is only 10 dB, this filter could be used as part of a
Direct-Form I IIR filter to generate zeros, while several pairs of poles could be used to
improve its stopband attenuation. A future implementation of this system would use this
scheme to improve the filter performance.

We have discussed the signal processing elements of the system. In the next chapter
the means of configuring them are briefly presented.

\textsuperscript{2}The idea was original but far from novel [32]!
6 USRP2 control and configuration

An important project goal was a flexible and easily reconfigurable system, which is important for a laboratory environment. This chapter discusses how the system is controlled using a Microblaze CPU, and how the HDL modules and external peripherals are configured.

6.1 Microblaze system

The Microblaze is closed-source, provided as an IP core with the Xilinx Embedded Development Kit (EDK). I used Xilinx Platform Studio (analogous to Altera SOPC Builder) to design the architecture of the CPU and its peripherals, and Xilinx Software Development Kit to write and debug C code for the CPU itself.

Peripherals

Figure 6.1 shows the system within Xilinx Platform Studio. Because XPS is GUI-based, the system is presented pictorially.

The system uses on-board RAM, though a future version may use the USRP2 SRAM. Two two-channel 32-bit general-purpose input/output (GPIO) peripherals are used to configure the FM generation and demodulation modules; see Ch. 4 and Ch. 5 for details of their outputs.

The JTAG boundary scan block on the FPGA is shared by the mdm debugger and ICON controller, which is interfaced to the ILA (integrated logic analyser) and VIO (variable input/output) blocks. The latter two are connected to external signals, such as the ADC and DAC buses; Xilinx ISE does not allow a Chipscope controller and CPU debugger to coexist unless they are part of the same CPU system. Finally the SPI block is externally connected to the USRP2 SPI bus, which is used to configure the clock manager and DAC.

A bidirectional SPI peripheral is interfaced directly to the USRP2 SPI pins, which are routed to the clock manager and AD9777 DAC.

CPU software

The GPIO and SPI peripherals were controlled by using Xilinx C libraries from within the SDK. The critical sections of the code are listed in Appendix B; the section up to Line 44 is responsible for setting the DAC to 1x interpolation/1x modulation mode (the clock manager section is currently not needed, but may be used in a future system to boost/decrease the clock speed and reduce latency).
6.2. Automatic boot-up

Upon power-on, the USRP2 FPGA is programmed by a 1600-gate Xilinx XC9572 CPLD. The CPLD reads bitstream data from an SD card, and programs the FPGA in slave-serial mode (the CPLD bitstream is stored on internal PROM).

Initially I planned to program the SD card directly with a custom bitstream, allowing the USRP2 to begin functioning immediately after power-on; this is extremely desirable in a real-world control system. The SD card required programming in raw mode, and I was only able to find GNU/Linux utilities to carry this out. I ran GNU/Linux in a virtual machine, and the utility appeared to function correctly — when the original Ettus Research bitstream was written to the SD card, the USRP2 booted without any problems. Whenever I wrote my own bitstream, it would not execute correctly; combinational paths switching LEDs on functioned correctly, while no sequential paths (such as blinking LEDs) appeared to operate.

The failure may have been due to a variety of HDL issues, but my main suspicion is the CPLD failing to shift the FPGA into execution mode. There are several FPGA-CPLD interface pins that require careful attention, which were not investigated in this project due to time constraints.
7 Error signal optimisation and low-bandwidth locking

In the previous chapters we have discussed the design and performance of the individual system elements. This chapter discusses how the error signal produced by the USRP2 was optimised and used to stabilise the laser.

7.1 Error signal optimisation

The system was connected according to Figure 3.1, minus the connection between the USRP2 output and the MOGbox. The photodetector and USRP2 outputs were viewed simultaneously on an oscilloscope, and the MOGbox was used to manually centre the laser frequency on the cooling transition $\nu_0$ mentioned in §1.1. The laser piezoactuator was linearly swept across the $\nu_0$ transitions of both rubidium-85 and rubidium-87, and the resultant data were used to optimise the error signal before switching on the controller. The precise frequencies of the various peaks in the photodetector spectrum were known, and were used to establish the $x$ axis in the MTS spectra below.

The primary aims in the optimisation were to maximise both the MTS error signal peak-peak amplitude and the slope on the zero crossing. Both of these lead to improved SNR. The first step was finely adjusting the optical alignment of the system; the MTS amplitude was improved by roughly half.

Modulation frequency

A tradeoff was made in the choice of modulation frequency. Higher frequencies allow wider demodulation passbands and thus a wider control bandwidth, however the limited modulation bandwidth of the AOM and $\sin(x)/x$ rolloff of the DAC reduce this benefit due to a weaker error signal and thus a lower SNR. Neglecting these effects, according to simulations done previously using (1.11) the optimal modulation frequency is 7.5 MHz for amplitude and 4 MHz for slope [6]. There was insufficient time to measure the dependence of SNR, amplitude and slope on modulation frequency, as the IIR filters required recompiling each time their coefficients were altered, however the figures of merit appeared to vary by less than 50%. 6.25 MHz was selected as a compromise.

Residual amplitude modulation

The sideband amplitudes were set to be equal using the 80 MHz FM signal on the spectrum analyser, and the phases were both set to $\pi/2$. The result was unmistakably an MTS error
signal, however it was quite asymmetric; the top lobe was significantly larger than the bottom lobe, which had several ‘kinks’. The three-sideband error signal equation (1.11), repeated below, was used to interpret this;

$$S(\Delta) = \frac{C}{2 \sqrt{\gamma^2 + f_m^2}} \left\{ \left( L_1 + L_{-1/2} - L_{1/2} - L_{-1} \right) \cos(2\pi f_m t) \right. $$

$$+ \left. \left( D_1 - D_{1/2} - D_{-1/2} + D_{-1} \right) \sin(2\pi f_m t) \right\} \quad (1.11)$$

As shown in Figure 1.4, the MTS signal is a sum of Lorentzian functions. Each sideband in the spectrum of the probe beam (after its passage through the vapour cell) generates four of the Lorentzian terms in (1.11); one creates the $L_1$, $L_{-1/2}$, $D_1$ and $D_{-1/2}$ terms, while the other is responsible for the remaining four. The amplitudes of the $L$ (in-phase) terms versus the $D$ (quadrature) terms are adjustable by altering the demodulator phase (as discussed in §5.1).

Altering a sideband amplitude alters the amplitudes of its constituent Lorentzians; an unevenness in the sidebands is equivalent to the introduction of AM into the FM spectrum of the pump beam. This is known in the literature as residual amplitude modulation (RAM), and is caused by a variety of phenomena including imperfect overlap of the pump and probe beams, beam clipping due to poor alignment, and uneven modulation due to an imperfect optical modulator (the AOM in our system). It worsens the symmetry of the MTS signal, however heuristic techniques may be applied to cancel out one source of amplitude modulation with another. These can correct amplitude shifts in the concerned systems, however they fundamentally cannot alter sideband phase — independent alteration of the sideband phases is believed to be a key advantage of the FM synthesis approach in this project.

### Sideband and demodulator adjustment

The MTS spectra were collected using a Tektronix DPO-3014 digital oscilloscope. The outputs from the photodetector and the USRP2 were displayed. The frequency spacing between the peaks in the saturated absorption spectrum (from the photodetector) were known and used to calibrate the horizontal axes of the MTS spectra. The piezo stack of the laser was swept using a sawtooth generated by the MOGbox; all the spectra show some ripple due to acoustic noise caused by the sharp jump in the piezo at the discontinuity. Spectra were smoothed with a 10-point moving average (each spectrum is 20 000 - 100 000 points).

Figure 7.1 shows the alterations of an MTS signal as its RAM distortion was corrected. Initially the sideband contributions are unequal, as shown in Figure 7.1a). After some experimentation, a protocol was established to correct distortion:

1. The sidebands amplitudes were equalised using the spectrum analyser, and both sideband phases were set to 4000;
CHAPTER 7. ERROR SIGNAL OPTIMISATION AND LOW-BANDWIDTH LOCKING

Figure 7.1: a) Unadjusted MTS, with demodulator phase 8B00\textsubscript{16}. The contributions from each sideband are uneven. b) The phase is altered to 6000\textsubscript{16}, and the 74 and 86 MHz sideband amplitudes are 0F\textsubscript{16} and 1F\textsubscript{16}; sideband phases are 4000\textsubscript{16} and 4B00\textsubscript{16}. The upper and lower traces are shifted by ±0.3 units to distinguish their shapes. c) The phase is altered to A800\textsubscript{16}, producing a signal ideal for laser locking.

2. One sideband was turned off, and the demodulator phase was adjusted until the signal appeared similar to the sidebands in Figure 7.1b). A symmetric ‘double hump’ corresponds to a phase of 0 in (1.11) (compare with the simulation, Figure 1.4). The amplitude of the observed signal and the CPU amplitude setting were noted;

3. The sideband was switched off and the other was switched on, and its amplitude and phase were adjusted such that the trace closely resembled the negative of the first sideband trace;

4. The second sideband was switched on; the signal was now symmetrical (the red line in Figure 7.1b). The demodulator phase was adjusted to obtain a signal resembling Figure 7.1c).

When used together with standard RAM-minimising techniques, this process reliably resulted in a symmetric MTS signal. Some iteration was required, especially for modulation frequencies of 5 MHz and above.
7.1. ERROR SIGNAL OPTIMISATION

Figure 7.2: Optimised MTS scan, with saturated absorption (directly from the photodetector) included for reference. The left half of the spectral features are due to rubidium-87, while the right half are due to rubidium-85. Settings were as for Figure 7.1(c). The laser was operating near the edge of its mode, thus its actual frequency did not correspond precisely to the frequency axis of the plot.

Once the sideband ratio was established using the method above, the sidebands and carrier could be scaled linearly to alter the amplitude of the FM being generated without destroying the MTS symmetry. A simple MATLAB script was written to predict what hexadecimal settings to use; this is listed in Appendix B.7.

MTS amplitude was nonlinear with respect to the carrier and sidebands, and was optimal (for a fixed RF power) when the sidebands were $\sim 6$ dB below the carrier. This is unsurprising, as the double-pass AOM arrangement imposes double the sideband amplitude of the original FM, producing a carrier and sidebands with equal amplitude. The product between the carrier and sidebands, given a total RF power, is maximised when the three are equal in amplitude; this was indeed observed.

An optimised MTS scan is shown in Figure 7.2. It compares favourably with Figure 1.2 but is operating at double the modulation frequency. The image in Figure 1.2 was obtained with a lowpass filter whose cutoff was 20 kHz rather than almost 6 MHz, explaining the slightly lower SNR. There is strong noise at 1600 MHz detuning, due to the edge of a laser mode. The noise could be removed by increasing the laser diode current, however this reduced the scanning range and the manifold of Figure 7.2 could not be entirely captured.
Figure 7.3: **a)** Spectra of the photodetector signal under various conditions; these were used to deduce the major noise source and SNR of the system. **b)** The 6.25 MHz beat predicted by (2.10) when the laser frequency \( f_c \) within \( \sim 2 \) MHz of \( \nu_0 \). The noise from 6.15 to 6.35 MHz is lowered in amplitude by the control system, discussed in §7.3. Spectrum analyser RBW = 3 kHz, VBW = 300 Hz.

### 7.2 Photodetector signal

Both the saturated absorption and the MTS signals in Figure 7.2 show visible noise. Figure 7.3 presents the photodetector spectrum in various circumstances; the closely related error signal spectra are discussed in §7.3. To interpret the spectra, first the dominant source of noise had to be identified.

#### Noise sources

Three main noise sources contribute to the photodetector output: shot noise, electronic noise, and laser amplitude noise. The following summarises how the amplitude noise was identified to be the culprit\(^2\).

Shot noise is due to the Poisson process of individual photons exciting individual electrons; unlike a constant electric current, where charge carriers are correlated, photon arrivals at the photodetector are uncorrelated. The shot noise limit is fundamental and cannot normally be reduced, however the SNR can be increased by increasing the power of the probe beam. The noise power is proportional to the square root of the signal power, and by increasing the probe beam from 300 \( \mu \)W to 600 \( \mu \)W it was clearly seen that the noise floor increased by 6 dB rather than 3. This eliminated shot noise as a candidate.

\(^2\)A true detective story often involves shot noise produced by the culprit’s weapon; unfortunately this experiment was not as obliging!
Electronic noise within the photodetector arises from several sources; chiefly the internal transimpedance amplifier and resistors as well as electromagnetic interference. The noise clearly fell between the off-resonant and the dark traces; thus electronic noise was not the limiting factor, although it was close. In the future the Thorlabs PDA10A-EL photodetector may be replaced by a quieter PDA36A.

This left laser amplitude noise, discussed briefly in §1.4. We see the implications of this in the next section.

Interpretation of photodetector spectra

The first notable point is the difference between the dark and off-resonant traces. The off-resonant trace was obtained when the laser frequency was far from any spectroscopic features, thus it was virtually unmodified by the vapour cell and any interactions therein. The signal is around 3 dB above the dark (laser turned off) trace, with several ‘bumps’ below 3 MHz but smoothness above. Because the photodetector signal is proportional to amplitude, these bumps represent amplitude noise; notably the largest is close to DC and there is a second around 2.2 MHz. The first is most likely $1/f$ noise; the second was puzzling until the laser transfer function was obtained (discussed in §8.3).

Next we compare the off-resonant and on-resonant traces. The first feature of the on-resonant trace is that it is 4-5 dB above the off-resonant trace; this was interpreted as being due to the saturated absorption peak on which the signal was centred (at 0 MHz in Figure 7.2) oscillating upwards and downwards due to the amplitude noise in both the pump and probe beams, instead of the probe beam only\(^3\).

The next feature is an axis of symmetry centred around 3.125 MHz, owing to the MTS process within the vapour cell. Any laser frequency noise at a frequency $f_n$ alters the phase of the MTS signal at $f_m \pm f_n$, which is demodulated into amplitude variation in the error signal at frequency $f_n$; this is the cause of the symmetry. Hence the frequency noise close to DC, which corresponds closely to amplitude noise in the off-resonant trace, is visible as a band of noise centred around 6.25 MHz. Note that noise at $f_n = f_m/2$ is encoded by the MTS at the frequency $f_m - f_m/2 = f_m/2$, which is precisely the axis of symmetry.

There are two unusual spurs at 1.25 MHz and 5 MHz. Their source was not identified during the project. It is possible that they are artefacts of the USRP2 FM generator; this could be pinpointed by altering the modulation settings of the pump beam and noting any variation.

Signal-to-noise ratio

The MTS signals around 4 MHz and from 5.75 to 6.75 MHz are significantly above the surrounding noise floor, and the SNR varies from 10 to 20 dB. Outside these regions, in particular from 4.5 to 5.5 MHz, no distinct signal is visible, indicating that the MTS signal is weaker than the amplitude noise. This implies the laser frequency noise is too weak.

\(^3\)It would have been useful to block the pump beam and test this hypothesis, as well as leaving the pump beam on while switching off its frequency modulation. This was not carried out due to time constraints.
from 750 kHz to 1.75 MHz for the MTS process to detect. Because laser amplitude noise is intrinsic to the laser diode, the SNR in these regions cannot be easily improved; the MTS error signal amplitude has already been optimised. A future system may include a differential noise canceller, in which a reference beam is split from the laser and directed onto a second photodetector whose signal is subtracted from the probe beam photodetector output [35]. The noise would be common to both, resulting in its cancellation by the subtracter (within the subtracter bandwidth), while the MTS signal would only be present on the second and thus remain un-cancelled. The limitation in such a system would be electronic noise.

In §7.3, we will see that the demodulated MTS signal shares the same SNR properties as the photodetector signal.

Ground loop noise

The optical table used for this project had a bare metal surface, and noise was introduced from this into the system whenever a grounded connector (such as the end of a BNC cable) made contact with it. Care was taken to avoid contact and the ground loop problem was minimised. A second loop was caused by the spectrum analyser, which was avoided by using an isolation transformer.

7.3 Closing the loop

Once the MTS signal had been optimised, it was straightforward to activate the controller. As mentioned in §2.3, the MOGbox controls the external cavity length of the laser and the diode current. The cavity actuator is a piezoelectric stack with a bandwidth of around 10 kHz, while the MOGbox current path has a bandwidth of 100 kHz. A transfer function of the current was obtained using a vector network analyser (VNA), and attempts was made to model the MOGbox using the MATLAB system identification toolbox. The results were poor, with increasingly poor agreement up to 100 kHz at which point the model was different from the measurements by 20 dB in amplitude and 180° in phase. The approach of creating a detailed Laplace-domain model of the system was abandoned due to time constraints.

As shown in Figure 3.1, the cavity feedback path uses a double integrator to achieve strong noise suppression at low frequencies, and the current path uses a single integrator. The paths can be activated individually. The following procedure was followed for obtaining a stable lock:

1. The piezo feedback was switched on with minimal gain. The gain was increased until oscillation was visible on the MTS signal, then reduced slightly⁴;
2. The current feedback was switched on with minimal gain. The piezo gain was again increased until the onset of oscillation, then reduced slightly;
3. The current gain was increased until the error signal began to oscillate, then reduced.

⁴Without current feedback, the piezo was unable to suppress frequency fluctuations caused by audio; the error signal behaved like a (very expensive) microphone!
7.3. CLOSING THE LOOP

Figure 7.4: Error signal spectra obtained using the MOGbox in conjunction with digital MTS. Frequency noise is suppressed up to 150 kHz, above which it is increased. Spectrum analyser RBW = 30 Hz, VBW = 10 Hz, with 10x trace averaging.

Performance

Error signal spectra were recorded for two different current gains; these are shown in Figure 7.4. The control bandwidth achieved using the MOGbox was 150 kHz for optimal gain settings, with 20 dB of frequency noise reduction at 10 kHz.

The on-resonance, unlocked signal is 5–20 dB above the off-resonance signal, providing a clear measure of the error signal SNR. This is less than was hoped, however is more than adequate for suppressing acoustic noise. Notably the optimally locked error signal closely follows the off-resonant signal below 70 kHz, indicating that the noise suppression was noise-limited below this and controller-limited above (the optimally locked error signal falls below the off-resonant signal for frequencies below 13 kHz, but this was most likely an artefact in the noise). There is a ‘noise bump’ at 200-500 kHz, falling beyond the range of the controller. The locked signal shows an increase in this noise, which is due to the noise spectrum in the stabilisation range being shifted to higher frequencies; this is an indication that the controller is performing correctly. When the current gain was raised past the optimal point, this noise grew larger still and began to worsen the lock performance.

There were spurs in the locked spectrum at 50 and 250 kHz. The MOGbox has an internal oscillator that runs at 250 kHz, which may have interfered with the error signal electronics, causing the latter spur. It is unknown what caused the 50 kHz spur, but it may have been electromagnetic interference; most other sources are ruled out due to its presence on the dark spectrum.
Qualitatively, the laser remained stabilised in the face of all acoustic noise propagated by air, such as clapping and yelling, however lightly tapping a metal object on the laser cover threw the system out of lock. It is likely that acoustic waves through the metal were both stronger in their effect and at higher frequencies than those generated in air; because the noise suppression falls with increasing frequency, these were likelier to dislodge the laser frequency sufficiently rapidly to throw off the control loop.

The laser was successfully left in closed-loop mode for several hours during a weekend, however it failed an overnight test, most likely due to frequency drift beyond the range of the actuators. This was probably due to temperature changes. More trials are needed to gauge the day-to-day reliability of the system; preliminary results are very promising.

### 7.4 System limitations

The main limitation to the performance of a perfect controller is the error signal SNR. This was fundamentally limited by the level of laser amplitude noise around the modulation frequency and the weakness of the MTS process itself. The digital hardware was not an important noise source, as the SNR of the photodetector and error signal are very similar (if the hardware was causing significant noise, the error signal SNR would be poorer). Potential strategies to reduce the noise were identified.

SNR notwithstanding, the true system limitation was the limited MOGbox control bandwidth of 150 kHz. This was sufficient to ensure resistance to temperature and pressure fluctuations and acoustic disturbance, but as seen in Figure 7.4 it did not cover the main region of noise extending to 500 kHz. Progress was made towards increasing the control bandwidth, discussed in the next chapter.
8 Diode current injection

8.1 Motivation

As seen in the last chapter, the MOGbox control bandwidth was limited to 150 kHz. The latency of the USRP2 measured in §3.4 indicated that its potential control bandwidth was at least 500 kHz, which could only be achieved by bypassing the MOGbox entirely and directly controlling the laser diode. Designing a suitable wideband controller required the transfer function of the laser and USRP2 — the MOGbox could not be used as part of the loop due to its limited internal bandwidth.

The laser frequency is dependent on the diode current, however a transient current of as little as 10 mA is enough to destroy the diode while it is running at full power. Thus a circuit is required that protects the diode from any such transients.

8.2 Injection circuit design and construction

One solution to this problem is to use a P-JFET [36] as shown in Figure 8.1. An input signal alters the JFET gate voltage, in turn altering the JFET drain-source impedance and changing the amount of current the JFET shunts to ground. A JFET is preferred to a MOSFET for its superior linearity; a Motorola 2N5460 was used. The four diodes D2-D5 clamp the JFET gate voltage to avoid damage, and the elements R1, R2 and C1 form a filter that compensates the capacitance of J1, D2 and D5 at frequencies above 1 MHz.

![Current injection circuit used in this project, adapted from [36]. The laser diode is modelled by a 20Ω resistor in simulations.](image)

Figure 8.1: Current injection circuit used in this project, adapted from [36]. The laser diode is modelled by a 20Ω resistor in simulations.
This circuit was initially constructed on a prototyping board, and was found to have a 3 dB point of 10 MHz, well above the requirements of this project. It was then constructed on a veroboard with an SMA input and output. The system transfer function was obtained using the circuit; one of the advanced requirements of the project.

8.3 Obtaining the laser transfer function

The circuit output was connected in parallel with the laser diode using an SMA input on the laser headboard. The injection circuit input was connected to the output port of a two-port vector network analyser (VNA; Array Solutions VNA2180) through a 40 dB attenuator; this was used to probe the small-signal behaviour of the system and minimise its nonlinearity. The VNA input was first connected to the photodetector output, to measure the combined transfer functions of the current injection circuit, laser and photodetector.

To obtain this transfer function, the laser was tuned to the side of the narrow saturated absorption peak at -200 MHz in Figure 7.2. Thus oscillations in the laser frequency resulted in oscillations in the photodetector voltage, which were measured by the VNA. The result is shown in Figure 8.2a).

Next the USRP2 error signal output was connected to the VNA in place of the photodetector, and the laser frequency was centred on the MTS transition. The same procedure was repeated; its results are shown in Figure 8.2b).

Figure 8.2a) shows several interesting features. The system gain rolls off gradually; on a logarithmic plot the rolloff is -7 dB per decade. At 2.3 MHz there is an amplitude and phase shift; this is due to the internal physics of the laser diode and must be taken into account in the design of a controller. The transition is also the cause of the 2.3 MHz noise mentioned in §7.2, which is visible in Figure 7.3a).
Figure 8.3: Transfer function of the USRP2 alone; obtained by subtracting Figure 8.2a) from Figure 8.2b). A $2\pi$ phase shift occurs at 2.5 MHz. There is a slight dip at 2.3 MHz due to imperfect cancellation of the laser amplitude/phase shift.

Figure 8.2b) demonstrates the MTS system as a whole. The modulation frequency was 5.299 MHz, and the IIR filter passband was set to 3 MHz. The resultant amplitude resembles the saturated absorption trace above 200 kHz, however the phase lag increases much more rapidly.

Figure 8.3 was obtained by subtracting the saturated absorption transfer function from the MTS transfer function, thus isolating the transfer function of the USRP2. The amplitude is reasonably level below 3 MHz, verifying the smoothness of the IIR filter passband. The latency of the USRP2 causes a $2\pi$ phase shift at 2.5 MHz; this corresponds to a latency of $1/(2.5 \times 10^6) = 400$ ns. The latency measured in §3.4 was 350 ns, and as expected the demodulation logic contributed 5 clock cycles of delay. The majority of the latency, 260 ns, remains due to the AD9777 DAC despite it being in its most basic mode.

Based on the data above, a plan was made to modify the IIR filter cutoff to 400 kHz to shift the system 0 dB point down to 600 kHz, however the modifications caused instability. It is unknown why this was the case, but most likely due to excessive internal gain within the filters; there was insufficient time to make corrections.

Although the current feedback path was not used to successfully stabilise the laser, some preliminary tests were carried out. When the error signal was connected to the current injection circuit (through 30 dB of attenuation), the system locked after a fashion, although its oscillations were strong and uncontrollable. Based on the large phase shifts within the passband in Figure 8.2b), the gain and phase margins were -26 dB and -680°; thus it is unsurprising that the system did not lock! Nonetheless, the fact that the laser frequency oscillated at all is a sign that with some (admittedly heavy-handed) optimisation the current controller shall function correctly.
9 Conclusions and future work

This document has summarised the background theory, design and results obtained in the construction and optimisation of a digital modulation transfer spectrometer using the Ettus Research USRP2. A control bandwidth of 150 kHz was achieved, which was limited by the laser controller rather than the digital or optical hardware. FPGA-based signal processing techniques were developed and demonstrated whose performance was superior to analogue techniques in flexibility and reliability, and approximately equal in signal-to-noise ratio. An interface was developed that provided immediate access to critical parameters such as modulation frequency and demodulator phase, a vast improvement over standard analogue techniques in which these are adjusted manually using potentiometers, and a strategy of applying the new flexibility to create improved error signals was developed.

All of the basic project requirements were completed successfully, as well as several of the more advanced requirements. It was hoped to achieve a current injection-based lock in this project; although this was unsuccessful, the signal spectra and transfer functions discussed in the preceding two chapters provided detailed and valuable information on the system. The results will be applied in the near future to alter the IIR filters to achieve locking, as well as to create a detailed system model for simulations.

The main limitations of the project were hardware-related; the SNR of the MTS error signal was lower than expected, and the latency of the USRP2 DAC was 250 ns longer than expected. Despite these, the laser system is ready to be used in Dr. Turner’s laboratory; this was a major objective of this project. The laser will be fiber-coupled in the next few weeks at the time of writing (13/10/10) to provide a stable source of light for the laboratory experiments, several of which will be coming online in the next week. Additionally the novel FM generation and error signal optimisation techniques present an approach that as far as I have been able to ascertain has not been previously attempted; this is an unexpected bonus of the project. While digital laser stabilisation has been attempted previously, this appears to be the first time it has been applied to modulation transfer spectroscopy.

Overall, this project has satisfied the objective of creating a digital laser stabilisation system with performance equal or superior to an equivalent analogue system. The flexibility of the FPGA allows rapid modification of the digital signal processing parameters to suit new conditions, and there are no fatal shortcomings that cannot be bypassed to achieve control bandwidths of several megahertz.

Future work

The main remaining goal is to increase the system bandwidth to 500 kHz or more. This will include interfacing of a low-latency DAC and adjustment of the IIR filter transfer
function. Once this is complete the laser performance will be measured and compared with other locking schemes, allowing the strengths and weaknesses of digital modulation transfer spectroscopy to be identified. Modification of the FPGA bitstream to use purely open-source HDL will also improve the transparency and portability of the system. In the long term, the FPGA offers the potential of nonlinear and decision-based processing of the error signal; this could be used for example to automatically re-lock the system, or to increase error signal amplitude far from the zero crossing. These are just some of the areas where an FPGA-based system offers truly unique and unexplored potential.
A  Bill of materials

A.1  Optics

Optics are coated for 780 nm. Optical mounts, posts, post holders, screws, cage mounts etc. are not listed.

<table>
<thead>
<tr>
<th>Item</th>
<th>Model/part number</th>
<th>Quantity</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lenses, mirrors, beamsplitters and waveplates (focal lengths in mm)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F=100 plano-convex</td>
<td></td>
<td>1</td>
<td>L6</td>
</tr>
<tr>
<td>F=150 plano-convex</td>
<td></td>
<td>3</td>
<td>L2, L4, L7</td>
</tr>
<tr>
<td>F=-50 plano-concave</td>
<td></td>
<td>2</td>
<td>L1, L3</td>
</tr>
<tr>
<td>F=-15 plano-concave</td>
<td></td>
<td>1</td>
<td>L5</td>
</tr>
<tr>
<td>Flat mirror</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>10% beamsplitter</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Polarising beamsplitter cube</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Half-waveplate, low-order</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Quarter-waveplate, low-order</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Electro-optical</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Laser (custom-fabricated)</td>
<td></td>
<td>1</td>
<td>see §2.3</td>
</tr>
<tr>
<td>Acousto-optic modulator</td>
<td>Crystal Technology 3080-122</td>
<td>1</td>
<td>§2.2</td>
</tr>
<tr>
<td>Laser controller</td>
<td>MOGlabs DLC-202</td>
<td>1</td>
<td>Ch. 7</td>
</tr>
<tr>
<td>Photodetector</td>
<td>Thorlabs PDA10A-EL</td>
<td>1</td>
<td>§2.4</td>
</tr>
<tr>
<td>Test equipment (used for optical alignment)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power meter</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Beam profiler</td>
<td>Thorlabs</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>IR viewer</td>
<td>ADFADF</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>CCD camera</td>
<td>Thorlabs</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table A.1: Optical equipment/parts used in the project.
## A.2 Electronics

<table>
<thead>
<tr>
<th>Item</th>
<th>Model/part number</th>
<th>Quantity</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>USRP2-related</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Digital controller</td>
<td>Ettus Research USRP2</td>
<td>1</td>
<td>Ch. 3</td>
</tr>
<tr>
<td>RX daughterboard</td>
<td>Ettus Research LFRX</td>
<td>1</td>
<td>Figure 3.2</td>
</tr>
<tr>
<td>TX daughterboard</td>
<td>Ettus Research LFTX (modif.)</td>
<td>1</td>
<td>Fig. 4.1</td>
</tr>
<tr>
<td>JTAG programmer</td>
<td>Xilinx Platform Cable USB II</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td><strong>Signal processing</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF hi-power amplifier</td>
<td>Delta RF LA2-1-525-30</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>RF amplifier</td>
<td>Mini-circuits ZFL-500LN</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>50 MHz highpass filter</td>
<td>Mini-circuits BHP-50+</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td><strong>Test equipment</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscilloscope</td>
<td>Tektronix DPO3014</td>
<td>1</td>
<td>(throughout)</td>
</tr>
<tr>
<td>Spectrum analyser</td>
<td>Anritsu MS2721A</td>
<td>1</td>
<td>(throughout)</td>
</tr>
<tr>
<td>Vector network analyser</td>
<td>Array Solutions VNA2180</td>
<td>1</td>
<td>Ch. 8</td>
</tr>
</tbody>
</table>

Table A.2: Electronic equipment/parts used in the project.
B  Selected code listing

Verilog modules

Only hardware-implemented modules are listed below; testbenches and constraint files are omitted.

B.1 Main module

```verilog
//timescale 1ns / 1ps
//Vlad Negnevitsky, Apr 2010
//This module is embedded within a ‘wrapper’ that contains the original Ettus Research code
module usrp2_core(
  input iCLK_100, // 100 MHz clock
  input iRST_n, // global external reset
  input [13:0] iADC_A, //ADC pins, channel A
  input [13:0] iADC_B, //ADC pins, channel B
  output [15:0] oDAC_A, //DAC pins, channel A
  output [15:0] oDAC_B, //DAC pins, channel B
  input iSPI_MISO, // Master in slave out
  output oSPI_SCK, // SPI slave clock
  output [8:0] oSPI_SE_n, // SPI slave enables; TX DAC, CLK at bit 1, 0 of oSPI_SE_n
  output oSPI_MOSI, // Master out slave in
  // Output LEDs, including Ethernet LED
  output [5:0] oLED,
  // CPLD stuffs
  output oCPLD_START,
  output oCPLD_MODE,
  output oCPLD_DONE
);

//**********LED flasher**********
reg [27:0] clk_count;
```
assign oLED[4] = clk_count[27];

always @ (posedge iCLK_100 or negedge iRST_n) begin
    if (!iRST_n) begin
        clk_count <= 28’d0;
    end
    else begin
        clk_count <= clk_count + 1;
    end
end

wire [31:0] uP_vio;
assign oLED[3:0] = uP_vio[3:0];

wire [31:0] gpio0a_out;
wire [31:0] gpio0b_out;
wire [31:0] gpio1a_out;
wire [31:0] gpio1b_out;

wire [3:0] uP_spi_ss;
wire dsp_resetn = iRST_n & gpio0b_out[7];
wire comb_en = gpio1b_out[7];

DDS_3SIN fm_gen ( //sinewave FM generator
    .iCLK(iCLK_100),
    .iRST_n(dsp_resetn),
    .iFC(gpio0a_out[31:16]),
    .iFM(gpio0a_out[15:0]),
    .iGAIN_FL(gpio0b_out[31:24]),
    .iGAIN_FH(gpio0b_out[23:16]),
    .iGAIN_FC(gpio0b_out[15:8]),
    .iPHOFF_FL(gpio1a_out[31:16]),
    .iPHOFF_FH(gpio1a_out[15:0]),
    .oSIG_overflow(), //TODO: send this into an input GPIO later
    .oSIG(oDAC_B) //sent straight to DAC channel B (the one with capacitors removed)
);

// Demod processing
wire [15:0] demod_sig;

//Demodulator
demod demod1(
    .iCLK(iCLK_100),
    .iRST_n(dsp_resetn),
    .ISIG_IN(iADC_A),
    .ISIG_OFFSET(0),
    .iFM(gpio0a_out[15:0]),
    .iOSHIFT(gpio1b_out[15:12]));
APPENDIX B. SELECTED CODE LISTING

91 .iPHOFF(gpio1b_out[31:16]),
92 .oSIG_DEMOD(demod_sig)
93 );

//Frequency comb, for testing
96 wire [15:0] comb_sig;
98 freq_comb comb1 (  
99 .iCLK(iCLK_100),
100 .iRST_n(dsp_resetn),
101 .iFCOMB(gpio0a_out[15:0]),
102 .oSIG(comb_sig)
103 );
105 reg [15:0] IIR_stage1_in_r;
106 wire [15:0] IIR_stage2_in;
107 reg [15:0] IIR_stage2_out_r;
108 wire [15:0] IIR_stage2_out;
109 //Pipelining to reduce propagation delays
110 always @(posedge iCLK_100 or negedge dsp_resetn) begin
111 if(!dsp_resetn) begin
112 IIR_stage1_in_r<=0;
113 IIR_stage2_out_r<=0;
114 end
116 else begin
117 if(comb_en) begin
118 IIR_stage1_in_r<=comb_sig;
119 end
121 else begin
122 IIR_stage1_in_r<=demod_sig;
123 end
124 IIR_stage2_out_r<=IIR_stage2_out;
127 end
129 //IIR filtering
131 IIR_filter_stage1 filt1(
132 .iCLK(iCLK_100),
133 .iRST_n(dsp_Resetn),
134 .iU(IIR_stage1_in_r),
135 .oY(IIR_stage2_in)
136 );
138 IIR_filter_stage2 filt2(
139 .iCLK(iCLK_100),
140 .iRST_n(dsp_Resetn),
141 .iU(IIR_stage2_in),
142 .oY(IIR_stage2_out) //the channel with capacitors included
144 );
145 assign oDAC_A = IIR_stage2_out_r;
B.1. MAIN MODULE

// ******************************************************

// ************ Chipscope Interfacing ************

// Generate a 75 slower clock for the Chipscope ILA (note that this is BAD PRACTICE!
wire ila_clk_25;
assign ila_clk_25=clk_count[1];

// Trigger input to ILA
wire [7:0] ILA_trig;
assign ILA_trig={
    oSPI_SE_n[4:0],
iSPI_MISO,
oSPI_MOSI,
oSPI_SCK
};

// Data input to ILA
reg [95:0] ILA_data;

always @(posedge iCLK_100 or negedge iRST_n) begin
    if(!iRST_n) begin
        ILA_data=0;
    end
    else begin
        ILA_data={
            32’d0,
            IIR_stage1_in_r,
oDAC_B,
oDAC_A,
            2’d0,
iADC_A,
            gpio1b_out[1:0],
gpio1a_out[1:0],
gpio0b_out[1:0],
gpio0a_out[1:0],
            ILA_trig // bits 7-0 carry SPI data
        };
    end
end

// ***********************************************

// ***************** CPU interfacing ****************

up_controller u1 (  
    .iClk(iCLK_100),
    .iRst_n(iRST_n),
    .oVIO_async(uP_vio), // VIO output
    .oGPIO_0a(gpio0a_out), // FM centre and mod freq control
    .oGPIO_0b(gpio0b_out), // FM carrier and sideband amplitude control
APPENDIX B. SELECTED CODE LISTING

```plaintext
//GPIO outputs
.oGPIO_1a(gpio1a_out), // FM sideband relative phases
.oGPIO_1b(gpio1b_out), // Demod control

//GPIO outputs; currently unused
.oGPIO_2a(),
.oGPIO_2b(),

//SPI
.oSPI_SCK(oSPI_SCK),
.iSPI_MISO(iSPI_MISO),
.oSPI_MOSI(oSPI_MOSI),
.oSPI_SS(oSPI_SE_n), //TX DAC, CLK at bit 1, 0 of oSPI_SS

//Chipscope ILA
.iILA_clk(ila_clk_25),
.iILA_data(ILA_data),
.iILA_trig(ILA_trig)
);

//**************************** Legacy CPLD control (need to use this later to boot CPU)
.assign oCPLD_START=1’d0;
.assign oCPLD_MODE=1’d0;
.assign oCPLD_DONE=1’d1;
endmodule

B.2 FM generator

Design is discussed in §4.1.

'timescale 1ns / 1ps

// Vlad Negnevitsky, June 2010

module DDS_3SIN(
  input iCLK,
  input iRST_n,
  input [15:0] iFC, // carrier freq
  input [15:0] iFM, // mod freq
  input [7:0] iGAIN_FL, // lower sideband gain
  input [7:0] iGAIN_FH, // higher sideband gain
  input [7:0] iGAIN_FC, // carrier gain
  input [15:0] iPHOFF_FL, // lower sideband phase offset
  input [15:0] iPHOFF_FH, // lower sideband phase offset
  output oSIG_overflow, // overflow warning
)
B.2. FM GENERATOR

```verilog
output reg signed [15:0] oSIG //output; modulated sine wave
);

reg signed [15:0] dds_fl_ph;
reg signed [15:0] dds_fh_ph;
reg signed [15:0] dds_fc_ph;

wire signed [15:0] dds_fl_sin;
wire signed [15:0] dds_fh_sin;
wire signed [15:0] dds_fc_sin;

wire [9:0] gain_sum = iGAIN_FL+ iGAIN_FH+ iGAIN_FC;
assign oSIG_overflow = (gain_sum>10’d255); //high whenever there is an
overflow in the assigned gains

wire signed [23:0] dds_fl_sin_sc = dds_fl_sin*iGAIN_FL;
wire signed [23:0] dds_fh_sin_sc = dds_fh_sin*iGAIN_FH;
wire signed [23:0] dds_fc_sin_sc = dds_fc_sin*iGAIN_FC;

reg signed [23:0] sig_wide;

always @(posedge iCLK or negedge iRST_n) begin
if(!iRST_n) begin
    dds_fl_ph=iPHOFF_FL; //lower sideband
    dds_fh_ph=iPHOFF_FH; //upper sideband
    dds_fc_ph=16’d0;    //carrier
    oSIG=16’d0;
    sig_wide=24’d0;
end
else begin
    dds_fl_ph=dds_fl_ph+(iFC-iFM);    //lower sideband
    dds_fh_ph=dds_fh_ph+(iFC+iFM);    //upper sideband
    dds_fc_ph=dds_fc_ph+iFC;     //carrier
    //Gains are 0 -> 255; individually adjustable - note that the
default iGAINS should not exceed
    //8’d255 when added together. oSIG_overflow checks this condition.
sig_wide = dds_fl_sin_sc+dds_fh_sin_sc+dds_fc_sin_sc;
oSIG=sig_wide[23:8];
end

end

//DDS lookup tables
dds_16b_lut dds_fl(
    .clk(iCLK),
    .sclr(!iRST_n),
    .phase_in(dds_fl_ph),
    .sine(dds_fl_sin)
);

dds_16b_lut dds_fh(
    .clk(iCLK),
    .sclr(!iRST_n),
    .phase_in(dds_fh_ph),
    .sine(dds_fh_sin)
)
```

APPENDIX B. SELECTED CODE LISTING

B.3 Demodulator

Design is discussed in §5.2.

```verilog
//timescale 1ns / 1ps
//Vlad Negnevitsky, Aug 2010
module demod(
  input iCLK,
  input iRST_n,
  input signed [13:0] iSIG_IN,
  input signed [13:0] iSIG_OFFSET, //offset added to input signal, to compensate for ADC offset
  input [15:0] iFM,
  input [3:0] iOSHIFT, //output left-shift
  input [15:0] iPHOFF, //phase offset
  output signed [15:0] oSIG_DEMOD
);

reg signed [15:0] dds_ph; //Local oscillator phase
wire signed [15:0] dds_sin; //local oscillator output

//Signed bit shift to scale the input
wire signed [17:0] sig_in_sc = (iSIG_IN)<<iOSHIFT;

//Multiplier
wire signed [31:0] sig_dds_product =
  sig_in_sc*dds_sin;

//truncate multiplication result
assign oSIG_DEMOD = sig_dds_product[31:16];

always @(posedge iCLK or negedge iRST_n) begin
  if(!iRST_n) begin
    dds_ph<=iPHOFF;
  end
  else begin
    dds_ph<=dds_ph+iFM;
  end
end

//Local oscillator
```
B.4 Pulse generator

Design is discussed in §5.4.

```verilog
module freq_comb(
  input iCLK,
  input !iRST_n,
  input [15:0] iFCOMB,
  output [15:0] oSIG
);

reg [15:0] phase;
reg [15:0] phase_old;

always @(posedge iCLK or negedge iRST_n) begin
  if(!iRST_n) begin
    phase_old <= 0;
    phase <= 0;
    end
  else begin
    phase_old <= phase;
    phase <= phase + iFCOMB;
  end
end

wire wrap_true = phase_old > phase; // true when phase has just wrapped to 0; once per period
assign oSIG = {0, wrap_true, 14'd0}; // signed; pulse on bit 14 every time the phase wraps
```

B.5 IIR filter biquad

Design is discussed in §5.3. Second biquad is identical, but with different coefficients. Coefficients are hard-coded to reduce latency (they were not declared as parameters due...
to difficulties with signed/unsigned multiplication).

```verbatim
//Vlad Negnevitsky, August 2010
module IIR_filter_stage1(
    input iCLK,
    input iRST_n,
    input signed [15:0] iU, //unfiltered signal
    output signed [15:0] oY //filtered signal
);

wire signed [15:0] U=iU;
wire signed [17:0] Y;

wire signed [31:0] cU = U<<14;
wire signed [31:0] R1_in = cU;
reg signed [31:0] R1;
reg signed [17:0] Yr;

wire signed [37:0] R2_in = bU+eYr+R1;
reg signed [37:0] R2;

wire signed [31:0] aU = U<<14;
wire signed [35:0] dYr = 28815*Yr;
wire signed [37:0] Y_full = R2 + dYr+aU;
assign oY=Yr[17:2];

always @(posedge iCLK or negedge iRST_n) begin
    if(!iRST_n) begin
        R1 <=0;
        R2 <=0;
        Yr <= 0;
    end
    else begin
        //propagate registers
        R1 <=R1_in;
        R2 <=R2_in;
    end
end
```
B.6 C program for Microblaze

The program was adapted from the Microblaze peripheral-testing demo code. Only a subset of the full program is presented for conciseness; for a complete listing please see web reference to project.

```c
// *** Subset of testperiph.c ***
// Vlad Negnevitsky, Aug 2010
// based on demonstration code by Xilinx, Inc.

// Data to send to the clock manager
clk_man_w_buffer[0]=0x204B8000;

// Sending procedure
TransferInProgress = TRUE;
XSpi_SetSlaveSelect(SpiInstancePtr,SPI_SS_CLK);
XSpi_Transfer(SpiInstancePtr,(u8 *)clk_man_w_buffer,
(u8 *)clk_man_r_buffer, SPI_CLK_MAN_BUFFER_SIZE*4);

// Wait for the transmission to be complete.
while (TransferInProgress);

// Acknowledge successful completion
print("CLK manager SPI transfer done.\n");

// Data to send to the DAC
// *** PLL off, 1x interpolation, 1x modulation
// *** (images at 20 MHz and 80 MHz)
dac_w_buffer[0]=0x44000000;
dac_w_buffer[1]=0x42000406;

TransferInProgress = TRUE;
XSpi_SetSlaveSelect(SpiInstancePtr,SPI_SS_DAC);
XSpi_Transfer(SpiInstancePtr,(u8 *)dac_w_buffer,
(u8 *)dac_r_buffer, SPI_DAC_BUFFER_SIZE*4);

// Wait for the transmission to complete
while (TransferInProgress);

// Acknowledge completion
print("DAC SPI transfer done.\n");

// ********************** Configuration of HDL blocks ******

// Initialisation
XGpio Gpio0;
```
XGpio_Initialize(&Gpio0, XPAR_GPIO_0_DEVICE_ID);
XGpio Gpio1;
XGpio_Initialize(&Gpio1, XPAR_GPIO_1_DEVICE_ID);
XGpio Gpio2;
XGpio_Initialize(&Gpio2, XPAR_GPIO_2_DEVICE_ID);

//------------- 6 MHz modulation settings -------------/
// 16b fc, 16b fm
XGpio_DiscreteWrite(&Gpio0, 1, 0x33331000);

// 16b lower sideband phase offset, 16b upper sideband phase offset
XGpio_DiscreteWrite(&Gpio1, 1, 0x48004B00);

// 16b demodulator phase offset, 4b demodulator left shift,
// (0x0 is unity gain, 4b nothing, 1b comb enable
XGpio_DiscreteWrite(&Gpio1, 2, 0xA8004000);

// 8b lower SB gain, 8b upper SB gain,
// 8b carrier gain, 1b reset_n, 7b unused
XGpio_DiscreteWrite(&Gpio0, 2, 0x00000000); // set reset_n low ;

// set reset_n high ---- sidebands are uneven,
// to compensate for the DAC filter
XGpio_DiscreteWrite(&Gpio0, 2, 0x1B0F2380);

MATLAB files

B.7 Script for FM parameter calculation

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
% sideband_hex_calculator.m
% Vlad Negnevitsky, 2010
%
% Calculates the required sideband amplitudes, in hex, for getting various
% lowerings or raisings from the '0dB' reference points. Ensures that the
% total sum of the gains is below 0xFF (i.e. 255), to avoid overflow.
%
% Convention here is sideband A to be above the carrier, B is below the
% carrier, C at 80 MHz (upconverted).
%
% Set reference amplitudes
% These amplitudes are the defaults used inside the CPU; 3 MHz
sb_A_default = hex2dec('4F');
sb_B_default = hex2dec('32');
sb_C_default = hex2dec('3C');

% These amplitudes are the defaults used inside the CPU; 6 MHz
sb_A_default = hex2dec('38');
sb_B_default = hex2dec('17');
sb_C_default = hex2dec('1E');

% Below carrier, carrier, above carrier default gains
B.8 Script for IIR biquad simulation

```matlab
% Calculate new setpoints for a change in amplitude
% Alter the below three parameters to calculate new sideband amplitudes
sb_A_gain = -3; % dB, 83 MHz
sb_B_gain = -3; % dB
sb_C_gain = -3; % dB, 77 MHz

sb_A_fact = 10^(sb_A_gain/20);
sb_B_fact = 10^(sb_B_gain/20);
sb_C_fact = 10^(sb_C_gain/20);

gain_factors = [sb_B_fact sb_C_fact sb_A_fact]

sb_A_new = round(sb_A_default*sb_A_fact);
sb_B_new = round(sb_B_default*sb_B_fact);
sb_C_new = round(sb_C_default*sb_C_fact);

new_gains = [sb_B_new sb_C_new sb_A_new]

if sum(new_gains) < 256
    disp('Gains are within overflow limits."
else
    disp('WARNING: Gains are outside overflow limits! Normalising..."
    atten = 255/sum(new_gains);
sb_A_new=floor(sb_A_new*atten);
sb_B_new=floor(sb_B_new*atten);
sb_C_new=floor(sb_C_new*atten);

end

disp('New gains are: (using CPU order)"
fprintf('%s %s %s %n', dec2hex(sb_A_new), dec2hex(sb_B_new), dec2hex(sb_C_new));
disp('Copy and paste this:"
fprintf('0x%02s%02s%02s80\n', dec2hex(sb_A_new), dec2hex(sb_B_new), dec2hex(sb_C_new));
```

% iir_quantised_test.m
% Vlad Negnevitsky, 2010
% Simulates the MTS 4th-order two-biquad IIR filter with simulation of
% binary quantisation issues. Can be used to compare the performance of the
% Verilog-implemented filter with MATLAB predictions. The transfer function
% of the filter is also obtained by sending through a linear chirp and
% analysing the FFT of the filter with its simulation.

clc; clear; close all hidden;

len=65536;

% Stimulus type: 1 for chirp, 2 for comb, 3 for square, 4 for ramp, 5 for sine
stim=1;
sig_period=1024; %used for all but chirp

if (stim==1)
  U_phase = 1/4/len*(0:len).^2;
elseif (stim==2)
  U=fix(sin(2*pi*(U_phase))*2^14);
elseif (stim==3)
  U=repmat([zeros(1,sig_period-1),2^14],1,len/sig_period);
elseif (stim==4)
  U=repmat([linspace(-1,1,sig_period/2) linspace(1,-1,sig_period/2)]*2^14,...
    1,len/sig_period);
elseif (stim==5)
  U=sin(2*pi/sig_period*(0:len))*2^14;
elseif (stim==1)
  title_string=sprintf('Stimulus vs response');
end

if 3MHz
  a1=16384;
  b1=-30274;
  c1=16384;
  d1=29787;%d1=29787;
  e1=-13717;%e1 =-13717;
elseif 6.25 MHz
  a1=29787;%d1=29787;
  b1=-13717;
  c1=16384;
  d1=-30274;%d1=-30274;
  e1=-32139;
elseif 12.5 MHz
  a1=16384;
  b1=-32139;
  c1=16384;
  d1=-30274;%d1=-30274;
  e1=-13717;
elseif 25 MHz
  a1=16384;
  b1=-32139;
  c1=16384;
  d1=-30274;%d1=-30274;
  e1=-13717;
% a2 = 16384;
% b2 = -32139;
% c2 = 16384;
% d2 = 31939;
% e2 = -16058;

% Section 1 constants: scaled up to 16-bit signed (a = c = 1 unscaled)
% a1 = 16384;
% b1 = round(-1.572673545296539*16384);
% c1 = 16384;
% d1 = round(1.844494806865143*16384);
% e1 = round(-0.85910601310489132*16384);

% Section 2 constants: scaled up to 16-bit signed (a = c = 1 unscaled)
% a2 = 16384;
% b2 = round(-1.8901009328265037*16384);
% c2 = 16384;
% d2 = round(1.9152523201115907*16384);
% e2 = round(-0.95742231996151639*16384);

% Output of stage 1
T = zeros(size(U));

% Output of stage 2
Y = zeros(size(U));

% Section 1 registers
R1 = 0;
R2 = 0;
Rt = 0;

% Section 1 wires
T_r = 0;
T_full = 0;

% Section 2 registers
R3 = 0;
R4 = 0;

% Section 2 wires
Y_r = 0;
Y_full = 0;

% Pipeline registers
Rp1 = 0;
Rp2 = 0; % currently unused

% Y/U = (a + bz-1 + cz-2)/(1 + dz-1 + ez-2)
% Y + dz-1Y + ez-2Y = aU + bz-1U + cz-2U
% Y = aU + bz-1U + cz-2U + dz-1Y + ez-2Y
% = aU + z-1( bU - dY + z-1( cU - eY ))

% Filter simulation
for k = 1:len
% Section 1 combinational operations; bit rounding simulated using
% fix()

T_full = fix(a1*U(k)) + fix(d1*T_r) + R2;
T(k) = fix(T_r/(2^2)); % T_r is unit delayed

% Section 1 sequential operations; carried out after combinational
% calcs are complete
R2 = R1 + fix(b1*U(k)) + fix(e1*T_r);
R1 = fix(c1*U(k));
T_r = fix(T_full/(2^14));

% Section 2 combinational operations; bit rounding simulated using fix()

Y_full = fix(a2*Rp1) + fix(d2*Y_r) + R4;
Y(k) = fix(Y_r/(2^2));

% Section 2 sequential operations; carried out after combinational
% calcs are complete
R4 = R3 + fix(b2*Rp1) + fix(e2*Y_r);
R3 = fix(c2*Rp1);
Y_r = fix(Y_full/(2^14));

% Pipeline registers; these are updated last
Rp1 = T(k);

end

figure;

% Following 2 lines plot subplots of U and Y
subplot(2,1,1); plot(U);
subplot(2,1,2); plot(Y);

plot(0:len,U,'r',0:len,T,'b')

title(title_string);

fft_U = fft(U);
tf_U_mag = abs(fft_U(1:len/2)).^2;
tf_U_ang = unwrap(angle(fft_U(1:len/2))) * 180/pi;

fft_T = fft(T);
tf_T_mag = abs(fft_T(1:len/2)).^2;
tf_T_ang = unwrap(angle(fft_T(1:len/2))) * 180/pi;

fft_Y = fft(Y);
tf_Y_mag = abs(fft_Y(1:len/2)).^2;
tf_Y_ang = unwrap(angle(fft_Y(1:len/2))) * 180/pi;

fft_pow = 10*log10(tf_Y_mag./tf_U_mag);

figure('Position',[400 100 1000 400]);

fft_freq = (2:2:len)*50/len;

% Calculate 180-deg point

[tf_phase = tf_Y_ang - tf_U_ang;]

[a180,i180] = min(abs(tf_phase + 180));

plot(tf_freq,tf_pow,'b',...
(2:2:len)*50/len,(tf_Y_ang-tf_U_ang)*60/180,'r',...
[0 10],[−60 −60],'g',...
[tf_freq(i180) tf_freq(i180)],[0 -80],’g’);
axis([0 10 -80 0]);
xlabel(’Frequency (MHz)’);
legend(’Magnitude (dB)’,’Phase (deg/3)’,’180 degree point’);
References


