A Frequency Offset Lock for High Field Imaging

David Burns

August 22, 2006

Abstract

The design and implementation of a tuneable frequency offset lock for the high field imaging of magnetically and optically trapped ^{40}K or ^{87}Rb atoms is discussed. The tuneable grating stabilized laser is optically mixed with a master reference locked to the species' atomic lines using a saturated absorption spectroscopy. Heterodyning occurs on an GaAs ROSA detector and the beat frequency is phase-frequency locked to a tuneable frequency source using feedback control of the piezo voltage and laser current. The lock has a 500 kHz bandwidth and a range of 6.4 GHz above and below the master reference. Optical line width measurements have yet to be completed for the offset lock, however, in principle the tuned line width can approach that of the master reference.

Contents

1 Acknowledgements **3**

2 Introduction **3**

3 Overview **3**

4 Hardware Reference **5**
4.1 AD9858 DDS and Eval Board 5
4.2 RCM3200 RabbitCore Module 8
4.3 New Focus 6013 Vortex Grating Stabilized Laser 8
4.4 Chroma-Matic Board 9

5 Software **11**
5.1 Outline 11
5.2 Dynamic C 12
5.3 Rabbit Control 13
5.4 National Instruments Lab Windows CVI 14
5.5 ADWin Sequencer V13.0 15
5.6 WOLF Rabbit Monitor 17
5.7 AutoConfig Generator 18

6 For My Successors **19**
6.1 Current Issues and Required Work 19
6.2 Upgrades That Would be Nice 20
6.3 Building Another Chroma-Matic 22
6.4 Contact 22

1 Acknowledgements

I would like to start by thanking everyone in the Cold Atoms Group at the University of Toronto for all their help this summer. Thank you Marcius Extravour, Lindsay Leblanc, David McKay, and Seth Aubin for all of the patient help in the lab. I admire and am inspired by your dedication. Thank you Alan Stummer for all your help with laying out and debugging the Chroma-Matic board, without you, this would not have been possible. Thank you Joseph Thywissen for your continuous trust and support throughout the entire project, it meant a great deal to me. Finally, thank you to the National Science and Engineering Research Council for providing the Undergraduate Student Research Award program and funding this work.

2 Introduction

The purpose of this project was to design and construct a frequency offset lock for the imaging lasers in the Thywissen Lab for resolving magnetically and optically trapped ^{40}K or ^{87}Rb atoms under high magnetic fields. The pre-existing apparatus included New Focus Vortex lasers which are locked to the species' atomic lines using a saturated absorbtion spectroscopy. The offset lock is required to tune a laser source with the Zeeman shifts of atomic transitions under high magnetic fields for the light scattering experiment.[5]

This document provides an overview of the design and various hardware components to serve as guide for operators or anyone continuing or expanding on this work. One of the requirements for this project was integrating the frequency control of the tuning laser into the existing experimental sequencer. A portion of this document is dedicated to explaining the use and structure of the code authored for this purpose.

3 Overview

The reference laser is locked to the desired atomic transition of either ^{87}Rb or ^{40}K using a saturated absorbtion spectroscopy.[4] The tuning laser is optically mixed with the reference on a polarizing beam splitter and aligned to single moded fibre ensuring that both beams are identically polarized. The heterodyned beat frequency

$$f_{\text{beat}} = |f_{\text{tuning}} - f_{\text{reference}}| \tag{1}$$

is detected using a HFD6180-413 ROSA[1] from Advanced Optical Components, a standard GaAs telecom product with a 10 GHz bandwidth.[1] Ap-

[1]ROSA: Receiver Optical Sub-Assembly

proximately 500 μW of combined balanced optical power is used. The sinusoidal output is digitized using a Micrel SY88953L limiting amplifier. The signal is fed through a pair of binary prescaler dividers with a combined modulus of 16 in order to reduce the frequency to levels detectable by the locking electronics.

The divided down beat frequency is phase-frequency locked to a tuneable frequency source generated by an Analog Devices AD9858 DDS[2]. The AD9858 includes a PLL frequency synthesis block housing a digital phase frequency detector configured to drive a pair of charge pumps. The charge pumps generate an error signal proportional to the phase-frequency difference of the DDS and divided ROSA signals. The error is integrated and filtered before being fed back as control voltages for the laser current (band pass) and piezo crystal (low pass) driving the laser frequency towards the desired offset and finally stabilizing it there. In this way we are using the tuneable laser as the VCO to replace a crystal oscillator in what would otherwise be a typical PLL circuit. The optics and electronics assembly is pictured in Figure 1.

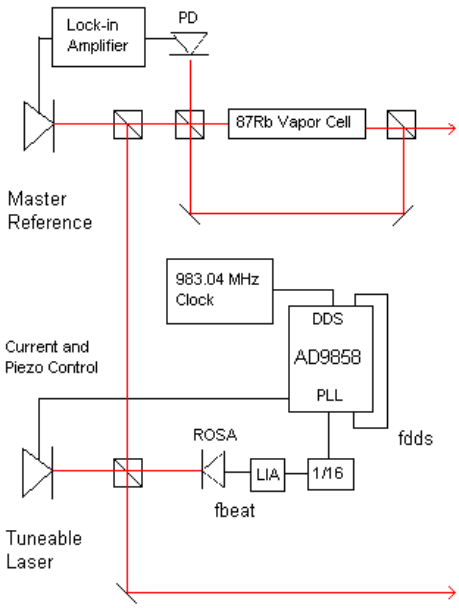


Figure 1: Offset Lock Optics and Electronics Assembly[10]

Real time frequency tuning is achieved by programming the AD9858 DDS over its parallel communications port. A Rabbit Semiconductor RCM3200

[2]DDS: Direct Digital Synthesis

microprocessor module is used for this task. Prior to running an experiment cycle, the series of commands required to produce the desired detunings are issued over TCP to the RabbitCore Module by the PC running our experimental sequencer, the *ADWin GUI*. The sequencer's primary purpose is programming the ADWin DSP Processor used to control the bulk of our electronics. The ADWin is additionally programmed to issue a series of trigger signals for the Rabbit used to synchronize the frequency tunings with the remainder of the experiment.

4 Hardware Reference

4.1 AD9858 PCB DDS Evaluation Board

The AD9858 PCB Evaluation board purchased from Analog devices houses the AD9858 DDS and PLL block. See the detailed hardware specification[2] and schematic.[3]

The DDS is used to generate a range of reference frequencies. The 983.04 MHz external clock is based on a Vectron OC0700-11-61.44MHz OCXO and was build by David McKay.[6] The DDS output frequency is set using a 32 bit integer dubbed the *frequency tuning word* (FTW). The DDS output can be set from 0-400 MHz in 0.25 Hz increments.

The beat frequency (offset) is divided by 16 on the evaluation board by a pair of binary prescaler dividers and compared with the reference frequency on the PLL block of the AD9858. The PLL block includes a phase frequency detector (PFD) used to drive the charge pumps (CP) between 0–5V. The baseline reference current for the charge pump is set using the loop gain potentiometer R_{LG} and can be calculated as follows

$$I_{\mathrm{CP}} = \frac{1.24V}{2.4k\Omega + R_{\mathrm{LG}}} \tag{2}$$

The integrated frequency error signal is used to drive the laser towards the desired offset frequency. Three programmable multipliers set the charge pump current as a function of the baseline reference current. The multiplier used is determined by the AD9858 locking logic as a function of the measured frequency difference. They are used in order of decreasing frequency error; Frequency Detect, Wide Closed Loop, and then Final Closed Loop. Note that during the Frequency Detect the charge pumps operate in an open loop mode where they are not driven by the PFD and simply output a constant current.

The PLL block on the AD9858 can be programmed to either increase or decrease the error signal for $f_{\mathrm{ref}} > f_{\mathrm{fbeat}}$. This is accomplished by changing the polarity of the charge pumps to ground or supply respectively. This allows the laser to be locked at a frequency above or below the master reference.

Both input stages on the PLL block contain an independently programmable binary divider supporting an additional frequency scaling with modulus 1, 2, or 4. The high frequency cutoff of PLL input varies with the divider modulus N; 150 MHz for N=1 and 400 MHz for N=4. This creates the absolute upper bound for the offset lock at 6.4 GHz. The divider modulus is adjusted with the offset frequency during sequence execution in order to avoid frequency cutoff at the PFD input. The lowest possible modulus is maintained at all times for greatest frequency resolution. The frequency offset can be calculated as follows

$$f_{\mathrm{offset}} = \frac{REF_{\mathrm{CP}} \cdot f_{\mathrm{DDS}}}{16N} \tag{3}$$

Where $\mathrm{REF_{CP}}$=1 for Gnd Ref or -1 for Sup Ref and

$$f_{\mathrm{DDS}} = \frac{FTW \cdot DDSCLK}{2^{32}} \tag{4}$$

The Chroma-Matic full lock range is 12.8 GHz centered about the reference laser. However, the full lock range cannot be fully traversed during a single experiment. This limitation is due to the fact that the charge pumps operate between 0V and 5V. Because the charge pumps saturate as they approach the rail voltages, they are in fact only operated between 1.5–3.5V. The piezo loop filter is unity gain, and hence, the piezo control voltage also only varies between 1.5–3.5V. The piezo crystal gain is approximately 1 GHz/V. Thus, the useable dynamically adjustable lock range is only 2 GHz. In order to access other frequencies, the piezo voltage setting must be adjusted manually on the Vortex Laser Controller. In principle, this limitation could be avoided by having the RabbitCore Module handle serial communications with the Laser Controller in order to adjust the piezo voltage setting, shifting the useable lock range, when the charge pumps approach saturation.

Control

The AD9858 supports external control in both serial and parallel modes. The RabbitCore microprocessor controls the DDS over 20 input pins in the faster parallel mode. In this mode, firstly, the DDS' 14 bit I/O buffer is loaded with the desired 6 bit Control Function Register (CFR) address and

an 8 bit command word. Changes incur following a "FUD" sequence which latches the contents of the I/O buffer into physical memory. A detailed register map is given in the documentation for the AD9858. [2]

The AD9858 stores four frequency profiles, each with its own FTW. The output is set using the FTW of the active profile selected using the P0 and P1 pins. Note that changing the active profile also latches the contents of the I/O buffer into memory similarly to a FUD sequence. In order to minimize response time for frequency changes and thus maximize timing accuracy, the FTW for the next f_{offset} in the sequence is stored into an inactive profile before the change is required. Upon receiving a trigger signal from the ADWin, the Rabbit need only trigger a profile change which involves setting 2 pins instead of roughly 64.

The AD9858 also supports a frequency sweep mode which allows linear sweeps in f_{offset} to be achieved with fine control over the ramping rate. However, unlike the AD9854[3], no feature exists for automating frequency sweeps between two specified values. A somewhat convoluted algorithm is required to achieve accurate timings for the ramp endpoints and an accurate destination frequency with the AD9858. The frequency sweep parameters are controlled by setting the delta frequency tuning word (DFTW) and the delta frequency ramp rate (DFRRW). During frequency sweeping mode, after DFRRW clock cycles the contents of DFTW is added to the frequency accumulator. The sum of the frequency accumulator and the FTW control the output frequency as the sum represents the step size for the phase accumulator.

Prior to initiating the ramp, the destination frequency is loaded into an inactive frequency profile and the DFTW and DFRRW are set. Additionally, a setting termed auto-clear frequency accumulator is enabled which resets the frequency accumulator to zero after a FUD sequence or profile change is received. During the ramp, the command to disable frequency sweeping is loaded into the I/O buffer. Upon receiving a trigger signal from the ADWin, the Rabbit only initiates a profile change. This simultaneously disables frequency sweeping, clears the frequency accumulator, and switches to the profile containing the FTW for the destination frequency.

Hardware Connections

Although the Evaluation board is certainly convenient as it includes all the

[3]The AD9854 DDS is used in the lab for generating RF sweeps for evaporative cooling of magnetically trapped atoms. It is not suitable for this application because it does not include a PLL synthesis block and operates at lower frequency.

necessary analog electronics for the AD9858, certain modifications were required for this project. Detailed instructions are located at `www2.physics.utoronto.ca/~astummer/mirror/Projects` under laser tuning.

4.2 RCM3200 RabbitCore Module

The RCM3200 Module from Rabbit Semiconductors is used to control the AD9858 DDS and several other Chroma-Matic board functions. The Rabbit firmware is coded in Dynamic C and loaded over an RS-232 programming cable. Control communications are made over tcp.

The RCM3200 is equipped with a 10/100Base-T Ethernet port, 52 parallel 5 V tolerant I/O lines, a real time clock and sports 512K flash memory, 256K battery-backed program data SRAM, and 512K program execution SRAM. The Rabbit firmware is typically stored in flash, however it can alternatively be loaded into program SRAM during development in order to avoid frequently writing to the flash and shortening its life cycle.[9]

4.3 New Focus 6013 Vortex Grating Stabilized Laser

The laser diode is tunable over an 75 GHz range (0.15 nm) centered between 710–800 nm. Piezo voltage and laser current are adjusted digitally with the use of a rotary knob on the front panel of the laser controller. Although not implemented this can be computer controlled over an RS-232 or GPIB interface. Three options are in place for frequency modulation, of which only two we use. The modulation inputs are on the back panel of the laser controller, and adjust the piezo voltage and laser current about the setting selected using the front panel or computer control interface. The characteristics are given below.[8]

DC Coupled Current Modulation	
Connector Type	BNC
Max. Voltage	$\pm$10V
Input Frequency Range	DC–1 MHz
Impedance	5kΩ
Modulation	0.2 mA/V
Frequency Gain	25–150 MHz/mA

Piezo Voltage Frequency Modulation	
Connector Type	BNC
Max. Voltage	±4.5V
Input Frequency Range	DC–3.5 kHz
Impedance	5kΩ
Frequency Gain	1 GHz/V

4.4 Chroma-Matic Board

The RCM3200 Module and AD9858 PCB are mated to the Chroma-Matic Board (CMB) layed out and assembled by Alan Stummer. The CMB powers and provides the interconnects for both and contains the PLL loop filters for both laser current and piezo feedback. The piezo feedback is low pass filtered with a cutoff at 3kHz for absolute frequency control and low frequency noise rejection. The current feedback filter passband goes from 3kHz to 500 kHz in order to narrow the tuned mode. The sum of the two filters yields a flat passband from DC-3kHz.[10]Loop gain is adjusted with the use of two front panel potentiometers. the first adjusts the charge pump reference current, affecting overall loop gain. The second controls the gain on the current filter.

The ROSA and LIA are mounted on a breakaway section of the CMB which is in turn mounted directly on the AD9858 PCB and connected to the prescaler divider input coupling.

The Chroma-Matic includes a National Semiconductor ADC124S101 4 Channel, 1 MSPS, 12-Bit A/D Converter[7] controlled by the RabbitCore Module. The primary ADC function is to monitor the charge pump voltage. Although not currently implemented, this could be used to generate warnings when the charge pumps approach saturation. The ADC is set up to monitor the loop gain and user potentiometer voltages. The former of which can be used to determine the charge pump reference current using Equation 2.

A schematic of the Chroma-Matic Board and DDS assembly are pictured in Figure 2. The labeled connections on the right of the figure indicate the inputs and outputs which are patched into BNC connectors on the rear panel of the assembly. Y-Axis carries the charge pump voltage and is monitored on the oscilloscope during scanning (described below). The signal is also displayed on an analog meter on the front panel. The piezo and current interconnects are connected directly to their respective rear panel control inputs on the 6013 New Focus Vortex laser controller box. Scope Out carries a square triggering signal for the oscilloscope during scanning. The sequence synchronization triggers for the Rabbit are received on the ADWin trigger input. The 3 pole 8 position switch is used to manually

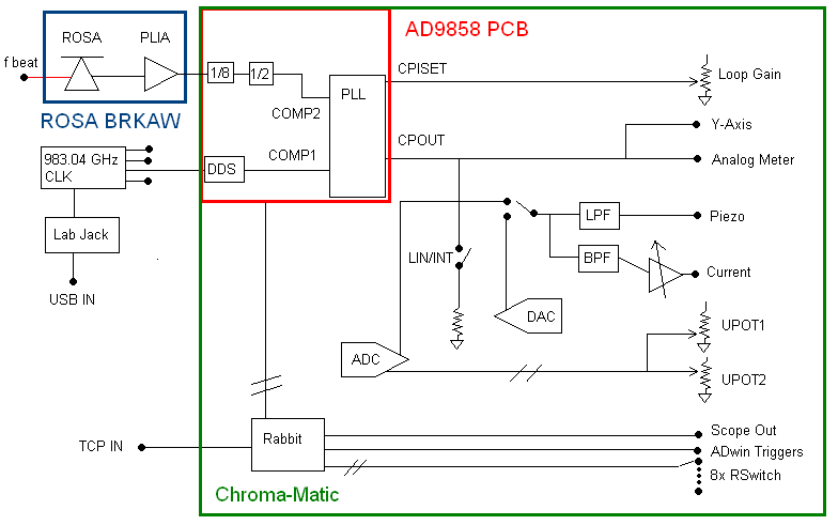


Figure 2: Chroma Board

run configuration routines saved in the Rabbit. UPOT1 and UPOT2 are potentiometers monitered by the ADC originally intended to provide an alternative to computer controlled frequency tunings. They are not currently in use.

It is often desirable, before experimentation, to scan the laser frequency over the used range and ensure a satisfactory mode is maintained. This is accomplished using one of two methods. The DDS can be scanned over the desired frequency range at a rate lower than the lock bandwidth. The locking circuitry drives the laser frequency to follow the DDS frequency scan. The integrated charge pump output, approximately proportional to the laser frequency, is monitored on an oscilloscope. Discontinuities in the output are evidence of mode hopping.

The second method involves disconnecting the piezo and current feedback from the charge pumps. Instead the filters are driven by a LTC1451 12 bit 5V DAC which is made to generate a triangular waveform. The DDS is held at a constant frequency. The charge pump current output is proportional to the difference between the beat and DDS frequencies. During normal operation, the voltage driving the loop filters is the CP current integrated across a capacitor. Instead, the integrated error is differentiated across the "linear" resistor. The resultant signal is proportional to the charge pump current and hence the frequency difference. It is displayed on the oscilloscope as it tracks the DAC scan, again giving an indication of the presence of mode hopping. The two CMOS switches in Figure 2 are controlled by the

RabbitCore Module and allow feedback to be switched from CP to DAC driven and frequency error to be switched from integrated to linear.

The picture given in Figure 3 depicts the partially completed assembly. Note that the ROSA Breakaway section is not pictured. Initially the ROSA was soldered directly to the AD9858 DDS Board.

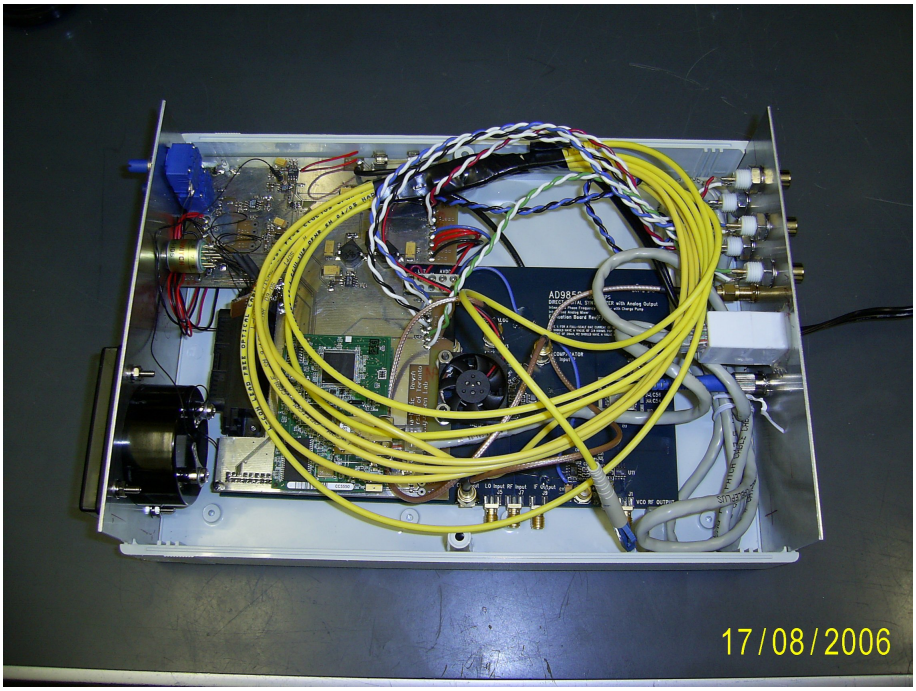


Figure 3: Laser Tuning Assembly and Box: The blue AD9858 DDS, green RCM3200 Module and tan Chroma-Matic Board. The yellow fiber patch cord transmits the optical signal to ROSA from FC barrel connector on rear panel.[10]

5 Software

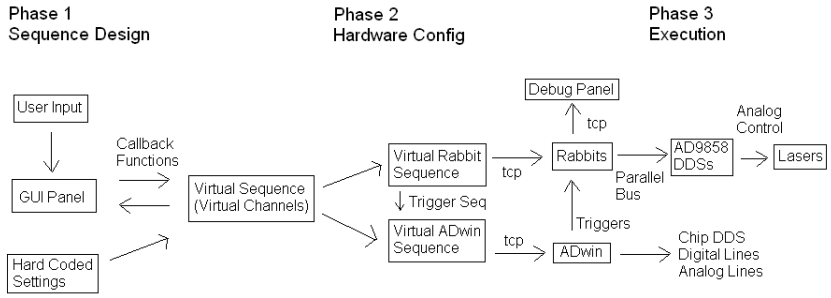
5.1 Outline

The software used to control the Chroma-Matic laser tuning project (and the rest of the experiment) is primarily split over three machines in three different languages. A PC running the ADWin GUI programmed in Lab Windows CVI is used to configure experimental sequences. The sequences are essentially divided into a number of time steps of controllable length. Within each time step, the experimental equipment is configured and held

in that state for the specified time period. A time line for each device is generated and displayed in a spreadsheet. Refer to figure 5. A virtual representation of the sequence is stored and altered through the use of callback functions called on user interactions with the GUI.

At run time the virtual representation of the sequence is translated into a series of commands which are communicated to our embedded devices. The ADWin houses 32 analog channels and two 32 bit digital banks used to control the bulk of the experimental electronics. The RabbitCore microprocessors are responsible for laser tuning. The ADWin also acts to synchronize all elements of the experiment. Temporal control of laser frequency updates is accomplished through trigger signals issued to the Rabbits by the ADWin. A schematic diagram of the software breakdown is given in figure 4, execution flows from left to right.

Figure 4: Software Virtual Breakdown



5.2 Dynamic C

Dynamic C is a C variant compiler designed specifically for the RabbitCore modules. Version 9.21 was used for this project. The following is intended for readers who are unfamiliar with its use and are required to modify the Rabbit firmware. For additional information refer to HTML documentation and sample programs.

During code development compile using the "Compile to RAM" option in order not to cycle the flash excessively. This loads the firmware into SRAM. Final versions should be loaded using "Compile to flash run in RAM" option. The program is then executed by powering on the Rabbit with the "Prog" cable removed.

Dynamic C (DC) does not support implicit referencing for path names of library and support files. Two steps are required to set the absolute paths correctly. Included with the source code is a template library file(template library.dir). Open this file in notepad and set the path names appropriately (hint: Ctrl-h for find and replace all). Secondly, select Option-Project Options from the DC pull down menu and go to Advanced on the Compiler tab. Check use user-defined template file and enter the path for the template library.

The DC compiler is dependent on comments within the source. The commenting format in use should be strictly observed, specifically, names of new functions must be added to the BeginHeader commenting block. Function descriptions are given at the bottom of the source and can be viewed by highlighting the function of interest and typing Ctrl-h.

5.3 Rabbit Control

The Rabbit's purpose is to program the AD9858 DDS during sequence execution in order to adjust the offset lock frequency as required for the experiment. Adjustments are triggered by the ADWin during runtime. The Rabbit software was designed to minimize the delay between triggers and realization of the desired adjustment (t_{adj}).

Commands are issued to the Rabbit by a PC over tcp. By default commands are executed immediately and in general, several commands are required to achieve a single adjustment of f_{offset}. This mode of operation is suitable for diagnostics or calibration.

Sequences of commands can alternatively be loaded into Rabbit RAM by prefixing each with an "ADDEVENT" header. The sequence of commands can then be executed in the order that they were loaded. A break command is used to divide the command sequence into batches. During run-time, the Rabbit executes all commands in a batch and then waits for a trigger signal from the ADWin before continuing with the next batch. The following is a typical example of an execution batch to adjust f_{offset}.

1. Load FTW into inactive profile
2. break
3. Adjust PFD Divider Modulus
4. Change Active Profile

Short sequences ($\lesssim$ 40 commands) can be transferred from RAM into a partition on the flash memory module (non-volatile). This is intended for frequently used calibration sequences. Space is allocated to store up to five "autoconfigs" in this manner. The autoconfigs can be recalled into RAM and executed at any time with a TCP command or using the rotary user switch.

At startup, the Rabbit module attempts to open an active TCP socket with the monitor console. If the connection is lost, the Rabbit can be reset or explicitly forced to re-establish the connection by connecting to the command socket and issuing the appropriate command. Error and status logs are communicated to the console following completion of an event sequence.

Software Structure

The Rabbit software was initially based on the structure of the DC sample program Echo Server, though it now bears little resemblance. The program loops about a simple state machine for the TCP command socket server. The states include Opening TCP Socket, Waiting for Connection, Connected and Receiving, Sequence Execution, and Closing Socket. General housekeeping functions, such as monitoring user switches, are multi-tasked in all states save the sequence execution state which consumes 100% of the processor time until completion. Triggers are received on a hardware interrupt line on the rising pulse edge. The interrupt callback counts pulses on a higher priority process independent of the main execution thread.

In the connected state, the Rabbit receives and parses incoming commands over TCP. The connection times out and is closed if no commands are received for a period greater than one second. The Rabbit also sends keep-alive packets every 500 ms in this state. Commands are identified by a header byte and are retrieved sequentially from the TCP stack. Invalid commands are logged.

The Rabbit software was optimized for speed and memory usage as a result of the module's very modest hardware. Note that floating point operation support is deliberately not included.

5.4 National Instruments Lab Windows CVI

LabWindows CVI is an ANSI C integrated development environment designed for instrument control, data acquisition, analysis, and user interface development. Version 7.1.1 was used for this project. The following is intended for readers who are unfamiliar with the environment.

The help documentation bundled by National Instruments is excellent so I am hoping to keep this brief and just outline the core principles and a few tips for creating and modifying VIs with Lab Windows.

Fortunately, Lab Windows automatically generates the bulk of the code for processing user and system events in the GUI. The programmer is required to design the GUI using a set of pre-made controls such as buttons and text boxes. Callback functions for the controls are automatically shelled out by Lab Windows and are called by the core process upon detection of a relevant event (ie. a mouse click on a button or a timer tick). The programmer is required to define the actions taken upon event detection before returning execution back to the core process. Because of this interrupt driven architecture, Lab Windows code is often very discontinuous. In my opinion one should begin understanding another's program by bringing up the user interface editor and matching callbacks with controls.

There is a massive user interface function library unique to Lab Windows which is used for controlling the GUI. Documentation is most easily accessed from the library tree toolbar.

5.5 ADWin Sequencer V13.0

The ADWin Sequencer, written in Lab Windows CVI, was originally designed to provide a GUI to automate programming the ADWin used for experimental sequencing and control. It was authored by Stephan Myrskog in *get date*, however, Seth Aubin, David McKay and I have all made additions to the code.

The ADWin controls a 5V 32 line analog bus, and two CMOS level 32 bit digital banks. The analog lines are generally used as control voltages, such as for adjusting AOM frequencies, or coil currents. The digital lines are generally used as triggers to operate shutters, etc.

The panel is organized into rows representing the analog and digital lines. DDS and laser tuning control is included at the bottom of the analog table. Each column represents a block of time, the amount is entered in ms above the analog table. The sequence is spread over ten pages, with one page that can be inserted at any point within the other nine. Note that a negative time value indicates that column should be skipped and the first 0 jumps the sequence to the next page. Within each column, analog values and DDS/laser frequencies can be stepped, ramped, or held constant.

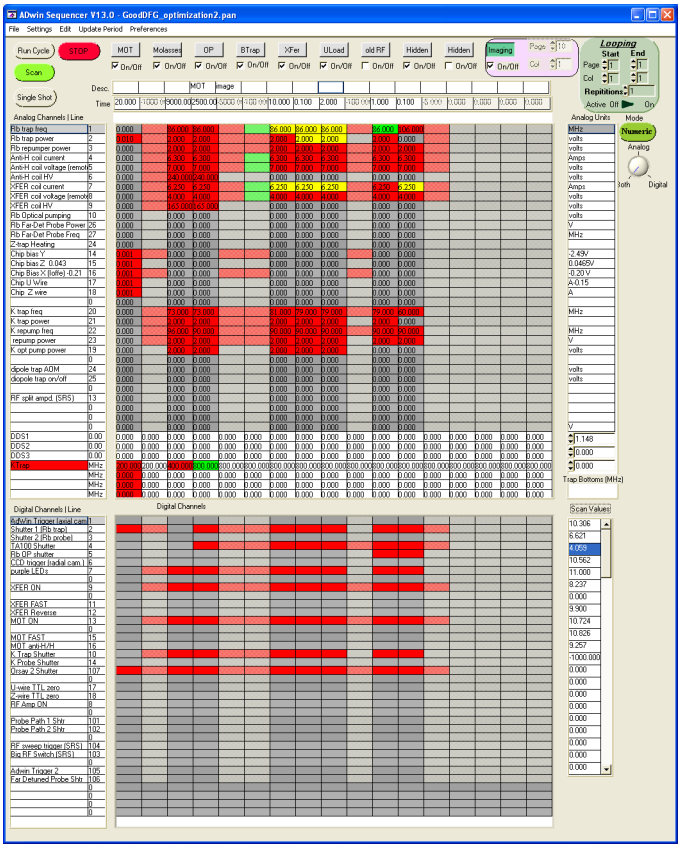


Figure 5: ADWin GUI Screen Capture

The sequence can be executed in one of three modes. Run executes the sequence a single time. Repeat repeats the sequence indefinitely until manually stopped. Scans allows a single parameter in the sequence (a column time, analog voltage, DDS frequency, laser frequency, etc.) to be varied across multiple, otherwise identical, runs.

Prior to execution, the laser tuning array is encoded as a string of TCP commands for the RabbitCore Module. The string is sent in 40 command bursts. Following each burst, the Sequencer waits for the Rabbit to parse, preprocess, and acknowledge the command string. The analog, digital, DDS, and Rabbit triggering arrays are then encoded as a series of updates and sent over TCP to the ADWin. Each column is broken down into a series of events, the period of which can be set between 5us and 1ms[4]. Each event consists of a series of updates which are executed sequentially. An

[4]10 μs is generally used, shorter event periods sometimes result in the DDS updates not finishing on time

example of an update would be changing the value of a single analog line, or updating one of the 32 bit digital banks. The event sequence is encoded in a set of three arrays. The first indicates how many updates are to be executed during each event. The following two arrays are synchronous and hold the update values and channels. Note that in this case "channel" refers to the code which the ADWin program uses to label its various I/O ports. For example an update on channel 101 would update the 1st digital bank with the 32 bit "value". This should not be confused with the labels used to identify the various digital and analog lines, which are sometimes also referred to as channels. A run command sent over TCP starts the ADWin program and experiment cycle.

Panels are easily saved and loaded, preserving all settings. The data is split over three files with the same name, differing only in their extension. Settings such as the event period are stored in a file with a .pan extension. Analog, digital, and DDS arrays are stored using a .arr extension. The laser tuning arrays and settings are stored using a .laser extension. Laser settings include DDS clock frequency, the IP address and port of the Rabbit Module's socket server, etc.

5.6 WOLF Rabbit Monitor

The WOLF Rabbit Monitor, written in Lab Windows CVI, displays and logs TCP traffic on up to four sockets, though this number could be easily extended. It is used to record error and status communications received from the embedded Rabbit microprocessors used to control the laser tuning project. A log file is created for each month and all entries are time and date stamped.

Connections are filtered by IP address. At startup it is required that the RabbitCore Modules' TCP settings be loaded in order to correctly set the filters. This is accomplished by loading a .laser file from a given panel save made with the ADWin GUI. The file is parsed for the relevant data.

The WOLF data sockets are opened in passive mode, they only accept incoming connections. However, if necessary, the WOLF can open a connection with the RabbitCore Module's command socket and send a command requesting for the Rabbit to open a connection to the data logging socket. This is accomplished using the "Force Connections" command button. The GUI is pictured in Figure 6.

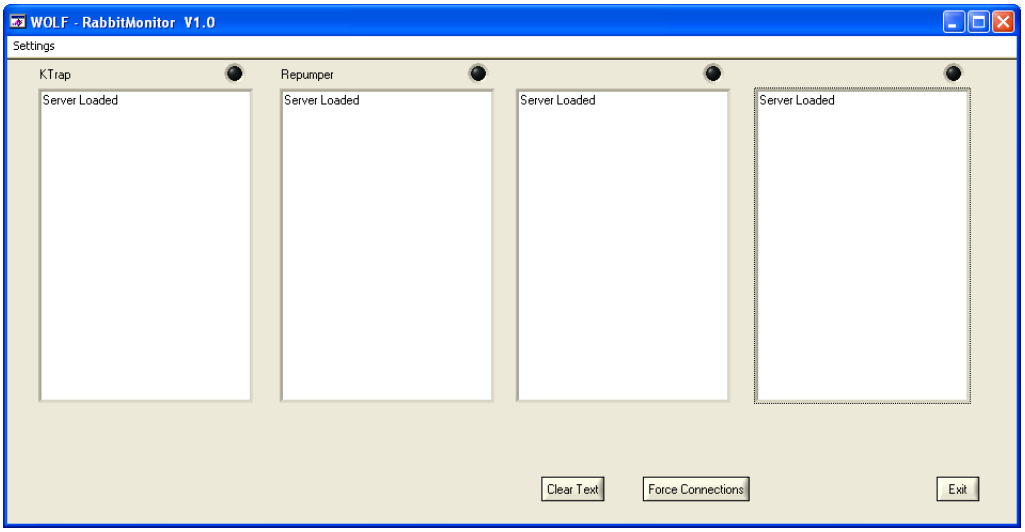


Figure 6: WOLF Rabbit Monitor Screen Capture: TCP Traffic is logged to text boxes and to file.

5.7 AutoConfig Generator

The AutoConfig Generator, written in Lab Windows CVI, generates simple command sequences for testing and running diagnostics on the laser lock. Sequences can be generated in order to lock the laser to a desired offset, or scan it using either the DDS or DAC scanning method. Refer to Section 4.4. A screen capture of the GUI program is given in Figure 7.

Prior to issuing the RabbitCore Module event sequences, the TCP settings for the devices must be loaded. This is accomplished exactly as with the WOLF Rabbit Monitor. The settings are loaded by selecting the .laser file from a panel saved using the ADWin GUI. After entering the parameters for the desired sequence and selecting the desired laser, the "Load Config to RAM and Test" command button sends the command sequence over TCP to the Rabbit and executes it once. If desired, the sequence can be saved to flash. The AutoConfig Index, values 0–4, denotes the location in flash memory the sequence will be stored. Sequences saved in this way can be executed using the "Test Config from Flash" command button or by selecting it using the rotary switch on the front panel of the Chroma-Matic Box. Saved sequence parameters are also saved to file by the AutoConfig Generator and displayed when cycling through lasers and index numbers.

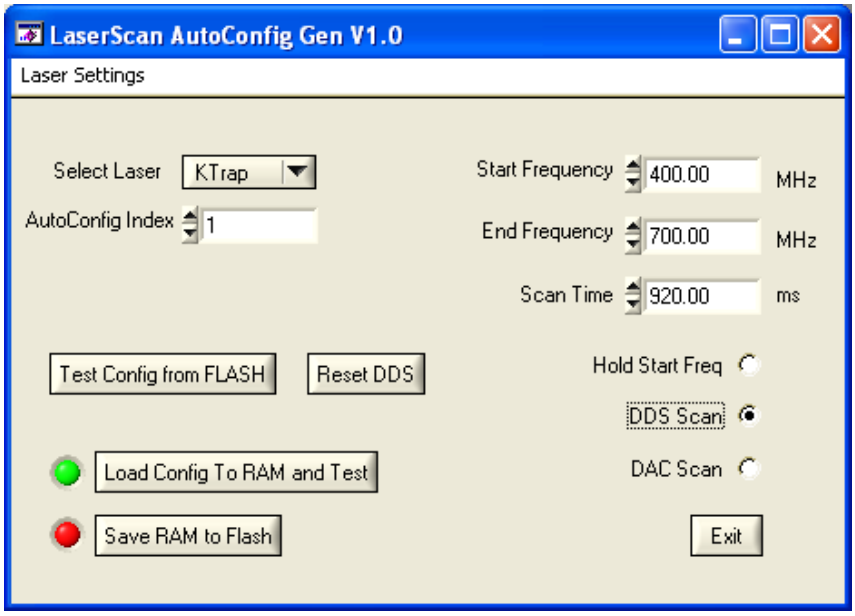


Figure 7: AutoConfig Generator Screen Capture

6 For My Successors

6.1 Current Issues and Required Work

I will begin by discussing the hardware additions and testing that are required before the Chroma-Matic laser tuning system can be integrated into the experimental apparatus.

During my last week, both AOC ROSAs were damaged, preventing final testing of the Chroma-Matic system. The cause was thought to be unregulated transient spikes from starting up the power supply used for testing the ROSA Breakaway board and Micrel LIA assembly. Three more ROSAs were ordered from AOC. The first step is to test the ROSA and LIA assembly on a safe power supply with mixed light and ensure that a digital beat signal is produced.

Preliminary testing of the locking circuitry with the ROSA output fed directly into the Hittite prescalers and piezo feedback only was successful, however, exceedingly noisy. The line width was not measured directly but was estimated to be 40 MHz. Electrical noise on the Chroma-Matic Board was thought to be the primary cause. The voltage to frequency gain at the piezo output is roughly 1 GHz/V so care must be taken to eliminate as much noise as possible. Several sources have already been isolated. The

initial design used a switching supply which turned out to be very noisy. The switching supply was replaced with a quieter linear regulator. Filtering the AD9858 cooling fan eliminated some noise as well. Unfortunately, the locking circuitry was not tested after these improvements or with current feedback.

If all measures of reducing electrical noise do not sufficiently narrow the mode, an alternative exists. The piezo feedback could be attenuated using a series resistance. This would of course reduce the range of the feedback voltage, and hence the usable lock range. However, this would also have the effect of scaling back the gain and reducing the the effect of noise.

The optical line width could be measured using the Fabry-Perot cavity, however, this would not necessarily provide a good measure of the lock stability. Any instabilities in the reference beam would inevitably be tracked and broaden the optical line width of the tuneable laser. Observing the ROSA or LIA output during locking provides a better indication of the lock stability. The electrical line width should be compared to that of the DDS signal or reference clock.

6.2 Upgrades That Would be Nice

Although the following additions to the project are not critical, I hope that at some point they could be addressed. The additions are all software related and are not overly difficult. I believe they would provide a good project for someone to to familiarize themselves with the software systems.

The Rabbit software currently does not include any protections against unauthorized connections. Although protected by the lab's firewall, I believe that an authentication routine would be appropriate. This could be implemented by adding an additional state which is entered after a connection is established but before entering a state where commands are parsed and executed. The state would begin by the Rabbit communicating an encryption code to the user which would subsequently encrypt and transmit the authentication password. A successful authentication would jump execution to the command parsing state and unsuccessful or timed out users would be disconnected.

The following addition would be very quick and simple. The WOLF Rabbit Moniter makes a log entry at startup, however, it does not on exit. This would be useful as it would provide a log of when the WOLF was running. Currently, if logs of successful sequence execution were found to be missing, it would be unclear if it was due to the WOLF being off, or due to some other malfunction.

Currently, the charge pump baseline current can be entered in the laser settings dialogue of the ADWin GUI. It is somewhat confusing because the entry has no other effect than updating the charge pump current display of the current multiplier settings. The current multiplier settings are in fact programmed into the DDS prior to sequence execution. As mentioned in 4.4, the ADC could be used to read the loop gain potentiometer voltage, and deduce the baseline current. It would be preferable if the ADWin GUI received this data from the Rabbit Module and displayed the baseline reference, as oppose to requiring it to be entered.

I will discuss the method used for the repeat and scan feature as it is currently one of the major limitations of the sequencer system. After sending the run command to the ADWin, a timer is initiated to the total sequence time plus some time alotted for error. For repeat mode, a timeout triggers what is essentially a call to the same algorithm used to run the first sequence, except that the ADWin is not reprogrammed. However, a significant amount of code executed between the timeout and sending the second run command to the ADWin. On a multi threaded PC running windows, code execution time can vary a great deal. This results in a varying cycle time for repeated sequences, a systematic that could be easily done away with. A more consistent cycle time would be achieved by timing the interval between sending run commands to the ADWin. The scan feature suffers from the same problem, however, it is compounded by the fact that the entire sequence must reloaded into the ADWin and RabbitCore Modules following the timeout. This added processing time increases error. I would recommend that the scan parameters be sent and stored on the ADWin and Rabbit as oppose to reloading the entire sequence each time. This would ideally be done in such a way that a multitude of parameters could be scanned simultaneously. However, the project would require modification to all three software systems; Sequencer, ADWin, and Rabbit, and would likely constitute a significant amount of work.

A "same as previous" cell type which does not update the value for its corresponding channel is useful because it allows a parameter to be scanned over multiple cells in adjacent columns. This feature was implemented for the laser tuning and analog cells, but not for the DDS lines. The reason for this being that the DDS structures do not include a parameter for cell type[5]. The addition of another parameter to the structure would make previously saved panels incompatible with the new code because the data arrays are stored in binary. This approach is possible, but a tool to convert the panels would be required. A similar tool was made for making version 11 panels

[5]Unlike the analog lines which allow for ramps, exponentials, steps, etc, a start and end frequency is all that is specified for each DDS cell

compatible with version 12. Another approach would be to create separate a array for the DDS mode and not include it in the structure.

6.3 Building Another Chroma-Matic

Assembling the hardware and setting up the optics would constitute the bulk of the work in building another system. All software was designed to support up to four laser tuning systems with little modification.

The only code modifications are networking related. Additional RabbitCore Modules would require a unique IP address and a port binding for the WOLF Monitor. These unfortunately must be hard coded in Dynamic C. The laser settings dialogue in the ADWin GUI allows for additional entries of laser tuning systems. The settings can then be loaded by the WOLF Rabbit Monitor and AutoConfig Generator.

6.4 Contact

Please direct questions to `mailto:3db14@qlink.queensu.ca`

References

[1] Advanced Optical Components. *10 Gbps 850nm PIN + Preamp Preliminary LC AND SC ROSA Package*, 2002.

[2] Analog Devices. *AD9858 1 GSPS Direct Digital Synthesizer*, 2003.

[3] Analog Devices. *AD9858 Evaluation Board*, 2003.

[4] S. Aubin, M. H. T. Extavour, S. Myrskog, L. J. LeBlanc, J. Esteve, S. Singh, P. Scrutton, D. McKay, R. McKenzie, I. D. Leroux, A. Stummer, and J. H. Thywissen. Trapping fermionic 40k and bosonic 87rb on a chip. *Low Temp. Phys*, 140.

[5] L. J. LeBlanc. *The Hyperfine Structure of Potassium-40*. PhD thesis, 2006.

[6] D. McKay. Microwave manipulation of atoms on a chip. *B.A.Sc. Thesis*, 2006.

[7] National Semiconductor. *ADC124S101 4 Channel, 1 MSPS, 12-Bit A/D Converter*, 2005.

[8] New Focus. *6000 Vortex Series Tunable Laser Diode*, 2002.

[9] Rabbit Semiconductor. *RabbitCore RCM3200 C-Programmable Module with Ethernet*, 2002.

[10] A. Stummer. Laser tuning. Technical report, University of Toronto, Quantum Optics Group, 2006. Available at `http://www2.physics.utoronto.ca/~astummer`.