Digital Improvements to an Ultracold Bose Gas Apparatus

Carlos Lopez-Abadia
Dept. of Physics, University of Colorado at Boulder

Defended April 6, 2018

Thesis Advisor:
Prof. Eric Cornell, Department of Physics

Thesis Committee:
Prof. Eric Cornell, Department of Physics
Prof. John Cumalat, Department of Physics
Prof. Steven Pollock, Department of Physics
Prof. Katherine Stange, Department of Mathematics

Table of Contents

Cover 1

Acknowledgements 3

Abstract 4

Introduction 5

Checks on Magnetic Field 7

Motivation 7

Solution 8

Setup 9

Frequency Response 10

Magnetic Field Detection 11

Spectroscopy Noise from Stray Magnetic Field Noise 13

Summary 16

Timing and File Synchronization 18

Motivation 18

Solution 19

Keeping Time on the Microcontroller 19

Saving the time 20

Software setup 21

Resource conflict 22

Operation 23

Control of Instrument Settings 25

Motivation 25

Solution 25

Conclusion 28

Bibliography 29

Appendix A-Feedback on microcontroller time 30

Acknowledgements

Firstly, I would like to thank the advisors I have had at CU Boulder: Eric Cornell, Jun Ye, and the late Deborah Jin. I cannot express enough gratitude to Eric for taking me in as a freshman and introducing me into the world of physics research. His guidance, patience, and example of leadership are invaluable. He continues to blow me away with his ability to jump between detail oriented and big picture thinking. Jun has an incredible eagerness to help, and I am fortunate to learn from his keen insight into complicated problems. Debbie gave me an amazing sense of encouragement when she sat down to work through problems with me as a freshman. I will remember that for the rest of my life.

I must thank graduate students Roman Chapurin, Xin Xie, Michael Van de Graaff, and Catherine Klauss. Roman is always astoundingly helpful and has particularly gone above and beyond in helping me prepare my thesis. I have known Xin since freshman year, and I am lucky to have known a graduate student with her patience and willingness to help for that length of time. Vandy is an excellent resource as well. He loves teaching and is remarkably good at boiling down concepts to build your understanding from the ground up. Cathy left before my work for this thesis but was a key resource for my earlier work. She also showed me a high standard for communication in science by example.

I also want to thank Jared Popowski, Bjorn Sumner, and Randall Ball. They are great friends also working in physics and have made my experience at CU Boulder much more enjoyable.

Lastly, I want to thank my family. My mother, father, and two brothers are an endless source of unconditional support. In particular, I want to thank my father for introducing me to science and sparking my interest in physics.

Abstract

Potassium 39 (K39) ultracold Bose gases provide an excellent opportunity to study few- and many-body physics. This is due to their rich Feshbach resonance spectrum, which allows for magnetically tunable interaction strength between the atoms in the gas. In our experiment we create such a Bose gas and tune a magnetic field to study it near a Feshbach resonance. Knowing this magnetic field is critical since it determines interaction strength. The complex experimental setup can benefit from improved automation and simplified data acquisition. In this thesis, I will outline my work towards realizing these goals.

Introduction

Ultracold Bose gases provide an excellent opportunity for the study of quantum many-body physics due to their magnetically tunable interactions strengths. The controllability of this interaction strength is due to magnetic field Feshbach resonances [1]. Accordingly, resonantly interacting Bose gases have attracted a great deal of attention [2-7].

In our experiment we create a K39 ultracold Bose gas and study it near a Feshbach resonance. We create this gas via a three-stage cooling process, with two magneto-optical traps (MOTS) and an optical dipole trap. Because tuning the magnetic field in this trap is what allows us to tune the interactions of the gas and study the resonances, knowing and controlling this magnetic field is of vital importance to the experiment.

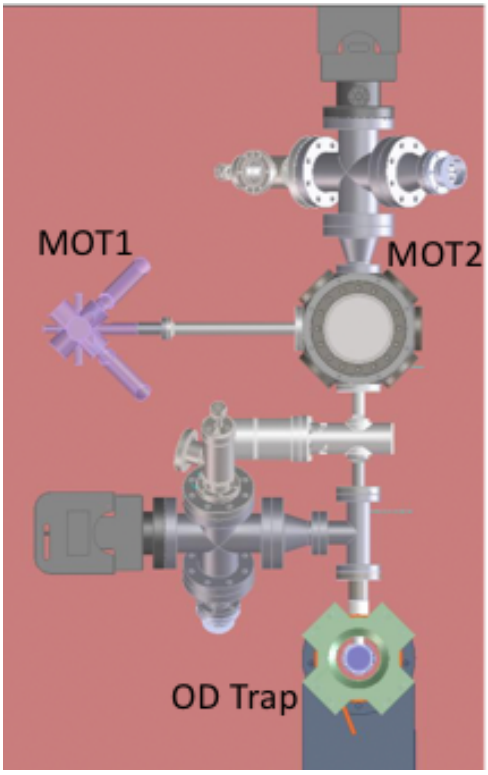

Figure 1: Diagram of the experimental apparatus. MOT 1 is the source of the cold atoms for MOT2. MOT2 performs Doppler and sub-Doppler cooling. A magnetic trap on a cart moves the atoms to the optical dipole trap, where they are cooled with evaporative cooling. Bias coils then generate the magnetic field used to control interaction strength between the atoms.

A set of current-carrying coils around the dipole trap, which I refer to as bias coils, create the uniform magnetic field used to vary the interaction strength of the gas. The magnetic field is controlled by the current we send through these coils. A typical experimental cycle involves setting interaction strength by changing this magnetic field to a specific value. Field-dependent spectroscopy is performed to determine magnetic field value and hence interaction strength. The atoms are then released and imaged at low field with the bias coils turned off.

The control for each of these stages of the experiment requires heavy use of software. Most of the experiment control is set by MATLAB scripts on lab computers that have an FPGA run the configured experimental cycle. An abundance of electronics are also used peripherally throughout this process to control lasers used for atom control and probing in the experiment.

With this multifaceted apparatus comes a lot of opportunities to improve how we acquire data, as well as how the electronics are controlled. In this thesis I document my work making use of some of these opportunities. In particular, I examine three improvements:

- Recording magnetic field near the optical trap as a secondary check of the field
- Providing accurate time-stamping for file synchronization
- Using software to control settings on electronic instruments in the apparatus

Checks on Magnetic Field

Motivation

Since we are controlling the strength of the Bose gas interactions through our magnetic field, knowing this magnetic field is very important to our experiment. This magnetic field is controlled by running a current through a set of coils we call the bias coils.

To determine the precise magnetic field generated by sending a certain current through these coils, we perform radio-frequency spectroscopy on the atoms in the Bose gas. While sending a fixed amount of current through the bias coils, we shine microwaves near the resonant frequency for the <F mF, 1 -1> to <F mF, 2 0> transition and measure how many atoms are later in the <F mF, 2 0> state as we vary the frequency of this light. By fitting the resulting data, we can find the center of the resonance, which is a well-known function of magnetic field [8].

While this is precise, it is also time-consuming. The experiment must be run once for each of the points to later be fitted. It takes around 20 points to determine the magnetic field to less than 1 mG uncertainty. With an experimental cycle of about 1 minute, it takes around 20 minutes to take this data. In an experiment where confidence in this magnetic field is important, it is useful to have a check on this magnetic field throughout.

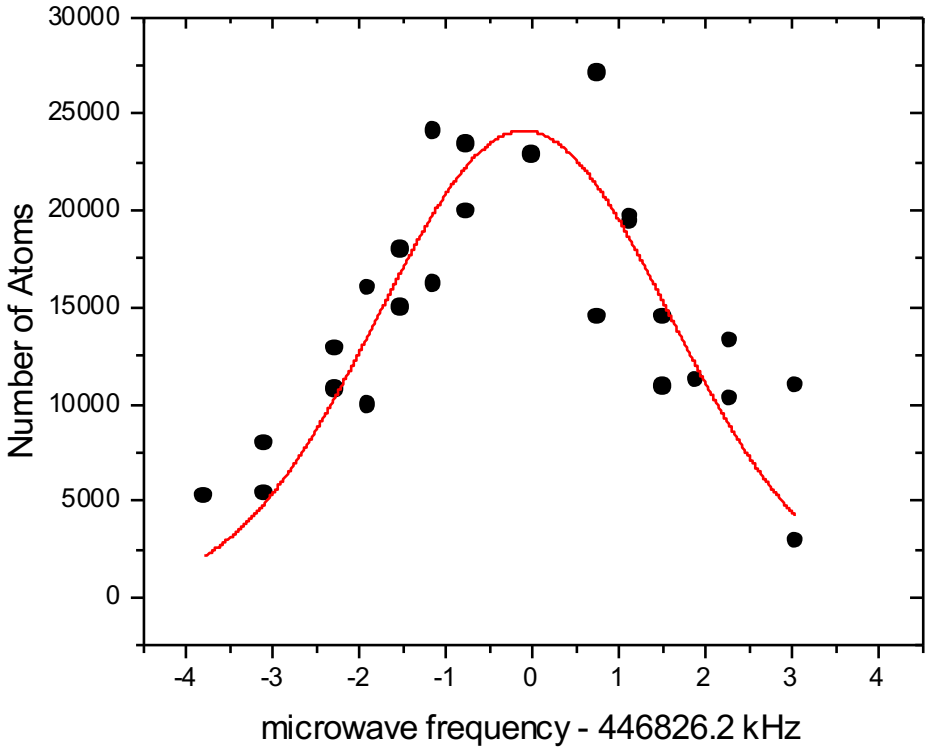

Figure 2: Sample atomic spectroscopy data. Each point is the number of atoms imaged in the Bose gas at a given frequency detuning. The center of the resulting fit gives the resonant frequency for the <F mF, 1 -1> to <F mF, 2 0> transition, which is a function of magnetic field. The fit is a Fourier transform of the functional form in the time domain of the pulse of microwave light shined on the atoms.

A check for the magnetic field must be defined by the following goals:

- Ability to detect magnetic field changes in the trap (the more precise the better)
- Trigger fast enough to synchronize with experiment (we want to know the time frame of magnetic field change that we are looking at)
- Easy integration into the experiment

Solution

To provide a magnetic field check that meets these goals, I set up an analog-to-digital converter (ADC) to take data when triggered from a magnetometer outside the optical dipole trap.

The resulting system has been easily integrated into the experiment and includes a digital trigger that we currently use to see the last 150 milliseconds of magnetic field change in our experimental cycle. Magnetic field changes caused far away from the experiment can be measured with a precision on the order of 0.1 mG. However, the frequency response of the magnetometer limits applications mostly to measuring the background magnetic field.

Setup

The ADC is an NI-USB 6003 Multifunction I/O device, and the magnetometer is a Bartington Mag690-1000 Three-Axis Magnetic Field Sensor. This ADC has good support in MATLAB, which is already used heavily throughout our experiment control.

The magnetometer installed outside the dipole trap, as shown in figure 3, sends magnetic field data to the data acquisition ADC, which logs the data to the computer via a USB connection. Limited access to the trap requires that the magnetometer is fixed to a breadboard above the trap. It is fixed at a position about 10 cm vertically and 10 cm laterally displaced from the trap. From this position it can sense background magnetic fields well, but not the exact field generated by the bias coils.

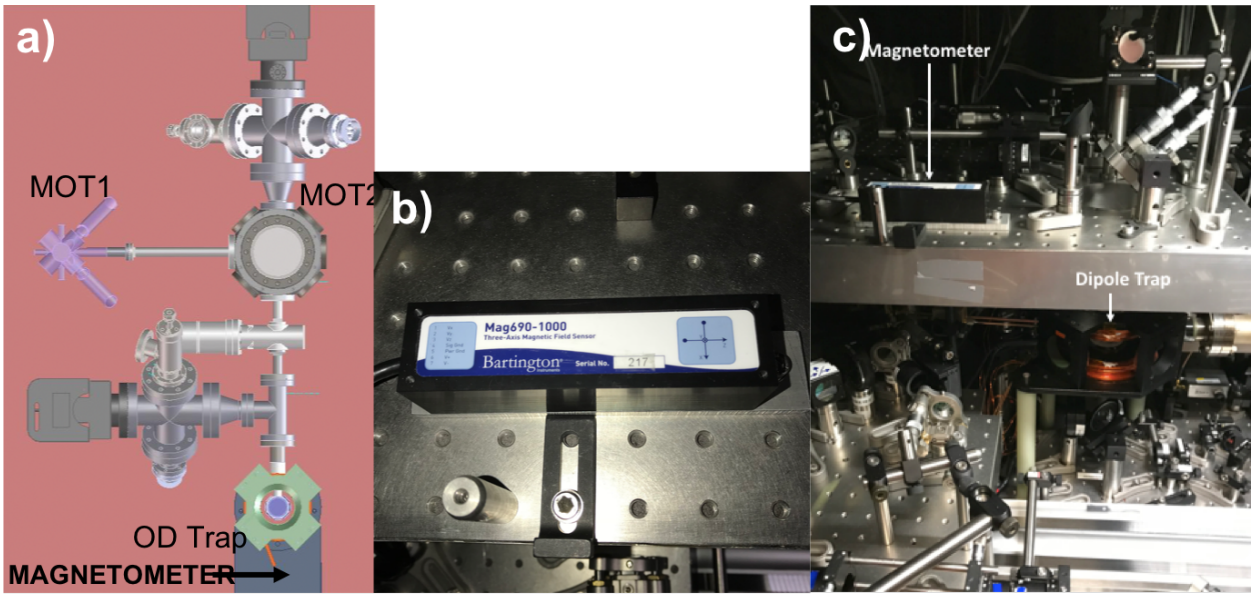

Figure 3: a) The experimental setup from figure 1 with the magnetometer position indicated. b) A close-up picture of the magnetometer. c) A picture showing the relative positions of the magnetometer and the dipole trap.

Data acquisition begins when an external digital trigger is sent to the ADC. The experiment control software is modified to trigger the device. Currently, we have this setup to trigger at the start of the last 150 ms of each experimental cycle.

A MATLAB program controls the ADC settings and writes the collected data to a text file as a four-column matrix, with one column for each of three axes of magnetic field measurement and one for time. The data files are saved to a folder based on their date and indexed by the order they were taken. As will be discussed later, following this standard is important for data integrity. Another MATLAB program can later read in this matrix for easy reading and manipulation of data.

Frequency Response

Figures 4 and 5 show the frequency response of the magnetometer and of the ADC, which have -3dB points of about 400Hz and 400 kHz, respectively. The ADC's sampling rate is

software limited to 100 kilo-samples per second (ksps), but nevertheless the limiting device is the magnetometer. The magnetometer's lower frequency fall-off limits the setup from detecting some of the faster changes to the magnetic field that are implemented during an experimental cycle.

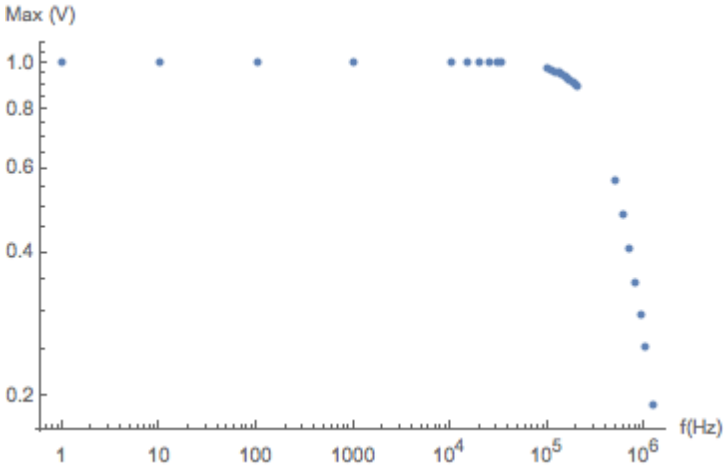

Figure 4: Frequency response of the ADC, measured as output for an input 1 V_{pp} wave of varying frequency. The half-power point is close to 400kHz. However, the sampling-rate is software limited to 100kHz, such that distinguishable signals will not be significantly attenuated.

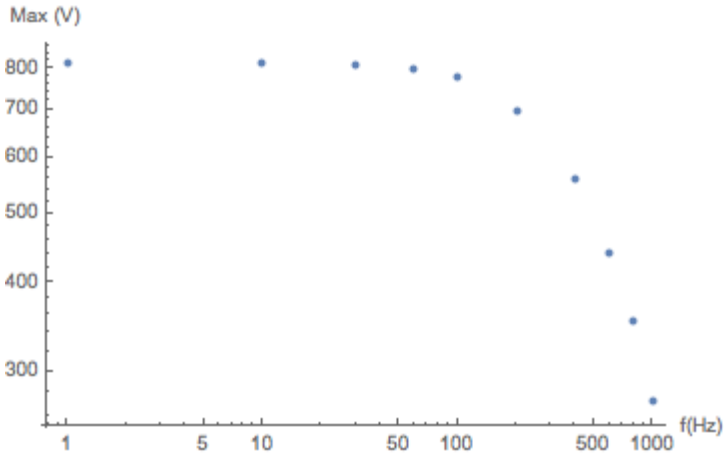

Figure 5: Frequency response of magnetometer, measured as output for an approximately 800 V_{pp} input. This was done using a test setup with a loop current-carrying wire with a measured radius, current, and number of turns. Half-power point is close to 400 Hz.

Magnetic Field Detection

For a source of stray magnetic field noise far away from the trap, a magnetic field change should be very similar at the magnetometer and at the center of the optical trap. The resolution of the magnetometer allows us to detect such shifts as small as 0.1 mG. This allowed us to detect shifts in the magnetic field on the order of 0.5 mG from the use of an elevator outside our lab.

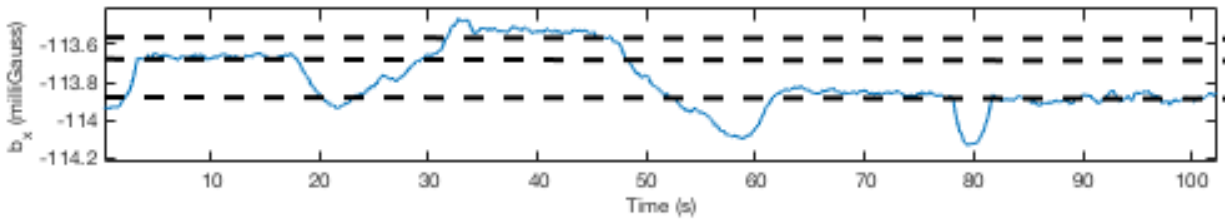

Figure 6: Magnetic field change along one axis due to elevator outside lab according to magnetometer. Elevator was ridden up and down over a 100 second period. Field plateaus indicative of elevator stopped at different floors. This demonstrates the resolution of the magnetometer setup.

Figure 7 shows the ability of the magnetometer to detect changes to magnetic field deliberately applied during the experimental cycle. Distinctive magnetic field changes that occur throughout the experiment are detected at the magnetometer. The magnetometer measures these changes to be around 2 orders of magnitude smaller than those occurring in the trap, measured along the same axis. This is because the magnetometer is well outside the axis of the bias coils that generate the field at the atoms.

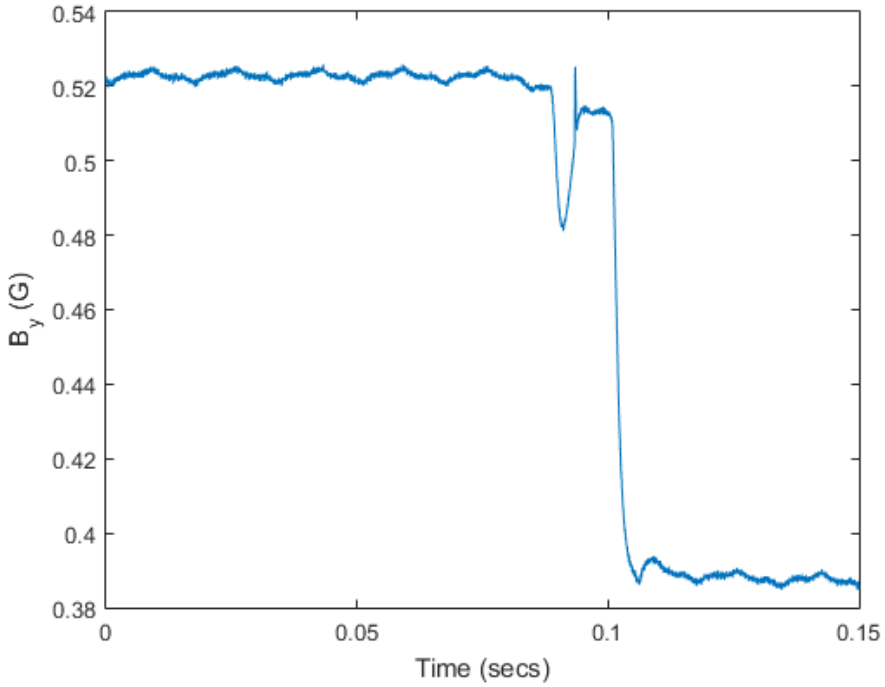

Figure 7: Magnetic field read by magnetometer along the axis of the bias coils over 150 milliseconds of experimental cycle when we manipulate the field. Clear features can be distinguished, like ramping and turning off of the bias field. The size of these measured magnetic field changes are around 2 orders of magnitude smaller than those inside the trap. Also visible are various harmonics of 60 Hz noise on the electronics. The experimental protocol involves some magnetic field changes with duration of about 1 millisecond, which is too fast for the bandwidth of the magnetometer to track.

This allows us to see what occurred in an experimental cycle. Visible are various harmonics of 60 Hz noise on the electronics. It is worth noting that this is noise in the electronics and not "real" noise from the magnetic field. The experimental protocol involves some magnetic field changes with duration of about 1 millisecond, which is too fast for the bandwidth of the magnetometer to track. The background magnetic field can also be calculated using the last 33.33 milliseconds of data in the cycle when the magnetic field is off.

Spectroscopy Noise from Stray Magnetic Field Noise

I conducted a test to see if the magnetometer can measure and identify the error contributed to spectroscopy by magnetic field fluctuations. I looked for correlations between deviations from the average background magnetic field on the magnetometer and deviations from the fit on atomic spectroscopy data. This process is shown in figure 8. The idea is that if a

significant part of noise on atomic spectroscopy data is due to deviations in the background magnetic field, then the residual values of the spectroscopy data (the value measured minus value predicted by the fit) will be correlated to the residual values of background magnetic field (the background magnetic field measured minus average background magnetic field).

The background magnetic field on each individual measurement is found by averaging over the last 2 periods of 16.67ms. This eliminates an observed 60Hz noise component from the signal and measures in a time interval in which the trap magnetic field is off so that only the background field is measured.

This analysis was performed only on data points on the steep part of the slope for each fit of atomic spectroscopy data. This is because in these points deviations in number due to magnetic field deviations should be more apparent.

Magnetic field sensitivity to frequency changes has a typical value of 0.15 MHz/G and varied from 0.1 MHz/G to 0.2 MHz/G in the data I examined. The slope on the spectroscopy fit is typically around 8,000,000 atoms/MHz. This gives an expected value of the slope of such a correlation of around 10^6 atoms per Gauss.

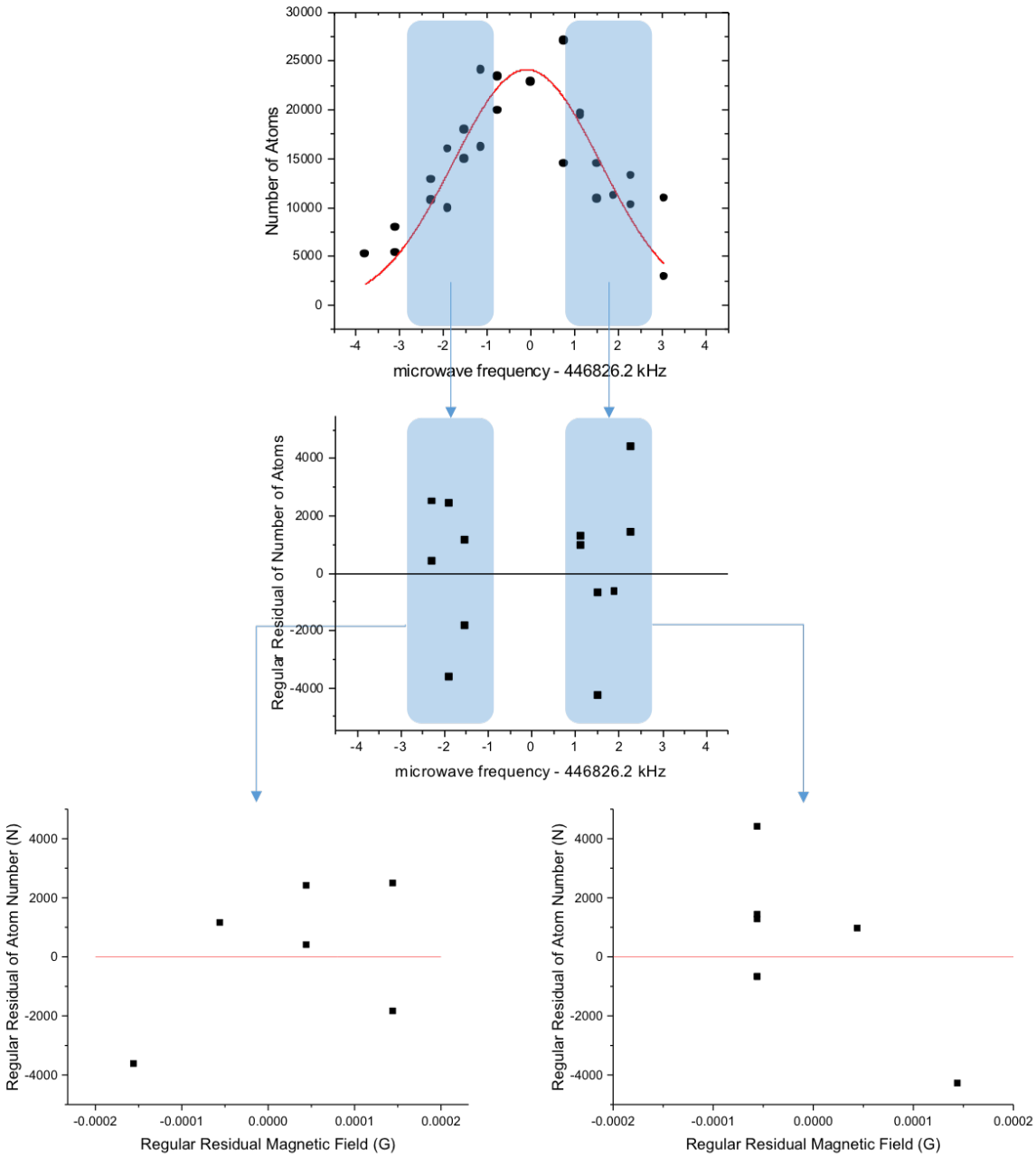

Figure 8: Process for analyzing data. The magnetometer gives the average background magnetic field of all points. The regular residual of this magnetic field with respect to the average magnetic field is calculated for each point. Then, points on the slope of the spectroscopy fit have their regular residual number values compared to the residual of their background magnetic field. These variables are examined for correlations.

I examined this relationship for twenty sets of spectroscopy data taken at ten different field values. I did not find any significant correlations in any of these data sets. Best fits of the data as well as fits to the line of expected slope had very low values for coefficients of determination (R^2). R^2 values were never more than 0.25 and the best fits were often off from the expected slope by orders of magnitude and even with the wrong sign for the slope altogether.

This result is most likely due to limited noise in the background magnetic field that we experienced over the examined spectroscopy data. With background magnetic field changes on the order of 0.2 mG, other noise sources, such as variation in the initial number of atoms in the Bose gas, obscure the effect of background field fluctuations. To be able to distinguish these correlations over 0.2 mG on typical atom measurements of 10,000 atoms, the number noise would have to be less than 2%. However, the data I analyzed had much larger number noise: between 5-10% (which is larger than the more typical 3% we often have). Performing this analysis on spectroscopy taken over more varied background magnetic fields may show less noisy data and significant correlations.

Summary

The limited frequency response of the magnetometer makes it difficult to derive significance from the measurement outside of the background field or the field when the experimental cycle begins.

However, the ability to measure this background field and the ability see the stages of an experimental cycle give the setup two main capabilities:

- Detection of changes to the background magnetic field in our trap
- Detection of failure of experiment electronics

Detection of a significant change to the background magnetic field helps prevent the collection of poor quality data. Background field changes above 0.1 mG are detectable, and changes of 0.3 mG introduce fluctuations in the number of atoms comparable to the 3% number noise we typically have. Data points with background field changes beyond a desired threshold,

let alone on a scale comparable to the bias field in the trap, can be thrown out to reduce noise on the data.

Detection of a failure of electronics or control systems helps to more quickly identify the source of a bad data point. If a certain experimental cycle fails completely because the magnetic field is not changed as expected, then inspection of the magnetometer data will show this. This can reduce debugging time if something is broken.

Timing and File Synchronization

Motivation

As we debug and run our experiment we may be interested in a variety of auxiliary data like the magnetometry already mentioned, but also temperature, humidity, laser frequency stability, etc... These measurements need to be correlated with the most basic tool for studying the atoms which is spatially and temporally resolved images of the atoms. In many cases the different kinds of data are collected by different software or even different computers and written to different storage.

A primary solution to this problem is choosing a standard for data association. In our lab the method we agreed to be most favorable is very simple: store data in folders based on the date it is taken, and number it based on the order it is taken on that day. Then matching auxiliary data to the image data or to each other is as easy as matching indices. However, when a set of data is skipped or missed, this causes a problem for synchronizing the remaining data.

A secondary measure to correlate files is needed and is provided by time-stamping data files. Still, the process of going through the computer's operating system to get this time-stamp can introduce a variable time of up to a few seconds added to the time when the data was taken. Additionally, the many different lab computers used may not be synchronized, and it may not be appropriate to synchronize them to a time server if they are offline. Depending on the system used to take the specific data, this delay can be considerably worse. For example, our image data can be recorded to the computer over 10 seconds after the image taking process is triggered.

While accuracy to 1 second on time-stamps is typically good enough for our experiment, any of the delays described above can place us outside this range. We need a method for time-stamping that can consistently provide accuracy to 1 second and preferably even less.

A solution to this problem has to be measured by the following goals:

- Consistent time-stamping of files
- Applicability to any data-taking system we have in the apparatus
- Minimization of dependence on lab computers

Solution

To meet these goals, I set up an Arduino microcontroller to maintain time externally from the computer. When triggered with a digital signal, the microcontroller saves the current time and then returns it to the computer for use as a time-stamp. Properly configured, this ensures that the time-stamp is accurate.

Keeping Time on the Microcontroller

The challenge of keeping accurate time on the microcontroller can be approached several ways. One method that was considered initially is to get the time through a GPS antenna in the setup shown in figure 9. The GPS antenna receives a digital GPS signal that encodes information about the current time [9].

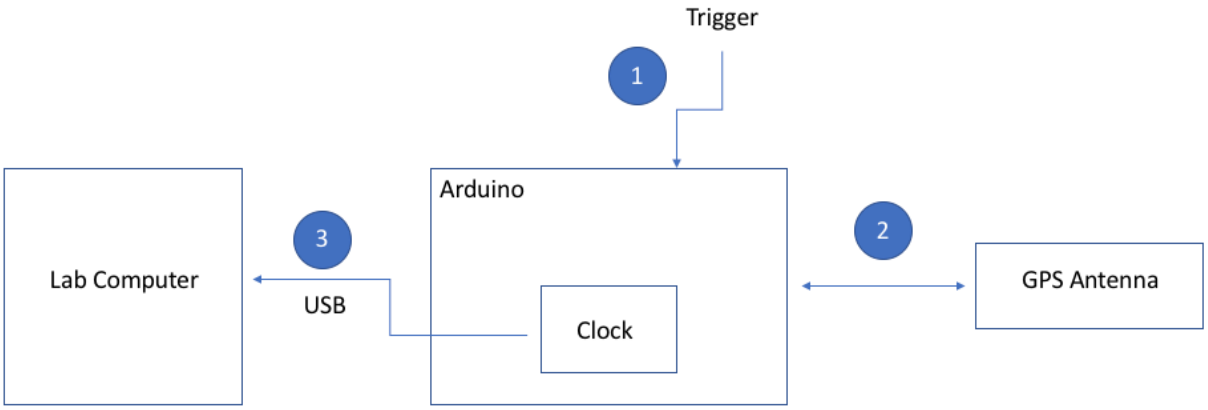

Figure 9: Possible design for microcontroller with synchronized clock. When trigger occurs at (1) the microcontroller retrieves the time through the GPS antenna (2) and sends it to the computer via USB (3). I did not implement this design because we did not have reception for the GPS antenna inside our lab.

While this design vastly reduces dependency on the computer, we faced a limitation in that the shielding of our lab prevents us from receiving a GPS signal inside the lab. Due to this, I

instead implemented the design shown in figure 10. In this configuration, the microcontroller's clock is synchronized with a lab computer that is synchronized to a NIST time server.

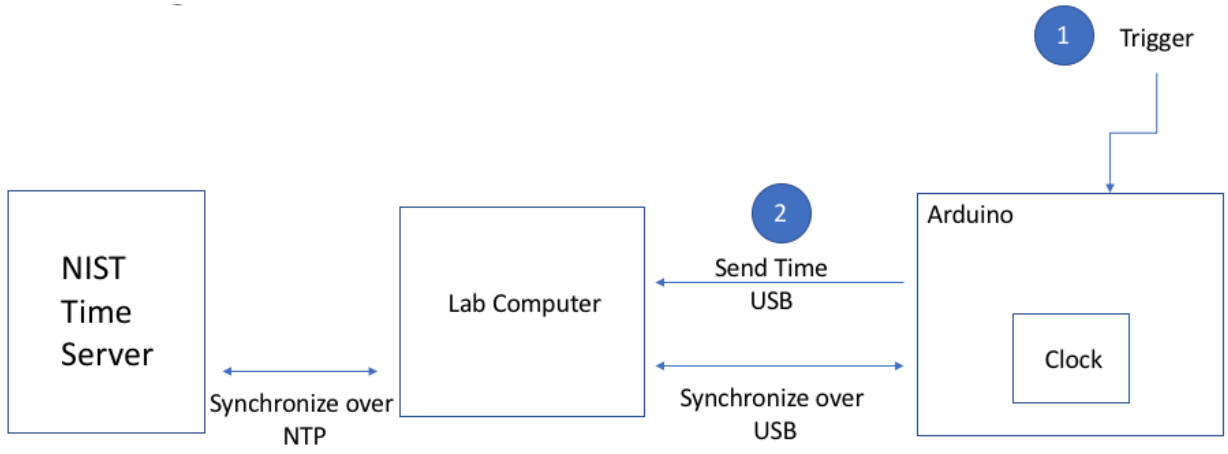

Figure 10: Design for microcontroller with synchronized clock. The microcontroller constantly synchronizes its clock to a computer that is synchronized to a time server. When trigger occurs at (1) the microcontroller retrieves the time from its clock and sends it to the computer via the same USB connection it uses for synchronization (2).

This introduces the dependency on the computer for proper synchronization of the microcontroller clock. I synchronized the microcontroller clock and then removed synchronization for 5 hours and 39 minutes and observed a delay in the microcontrollers time of 20 seconds to the nearest second. This implies the microcontroller's unsynchronized clock has a drifting rate of 0.059 ± 0.003 seconds per minute and can continue to provide time-stamps with its own clock that are accurate to within 1 second for almost 17 minutes. While this is very poor performance for a clock, the microcontroller is an inexpensive Arduino UNO. As long as feedback is provided when a disconnection from the computer occurs, this should allow us to maintain accurate time-stamps even with this computer dependence.

Saving the time

The time-stamp sent back to the computer is written on a new line of a text file. The times in this file will thus be ordered by file number of the data being taken. It is a simple task to match up time-stamps to the files as data is collected, regardless of the type of data or how it is taken. This also adheres to the standard we have laid out for file association. Retrieving these time-stamps should often be as easy as index matching.

I configured the microcontroller to send the time to this text file as a time_t data type, a type representing the number of seconds since 1 January 1970. This data type is particularly easy to use when setting a file's time-stamp in the operating system, as it is a standard used on Unix operating systems and there are many MATLAB functions that use this data type.

This choice limits the time-stamp's accuracy to within 1 second, which is good enough for most applications we have. It is simple to configure this to provide more precision, but the new time data needs to be interpreted accordingly on the computer side.

Software setup

The software to do this consists of a Processing sketch on the computer sending the time to a serial USB port connection while the microcontroller's software is constantly checking the serial port for the updated time. When this updated time is retrieved, it is used to set the microcontroller clock. A "DISCONNECTED" warning is issued when a check for synchronization is not passed. This is important to help ensure clock is re-synchronized in the window of 17 minutes that will still provide time-stamps accurate to within 1 second.

Importantly, the trigger makes use of a hardware interrupt on the microcontroller. This stops operation of the main program being run on the microcontroller and instead runs an interrupt service routine which gets the time and sends it to the serial port. This allows for quick retrieval of the current time when triggered, and hence more accurate time-stamps.

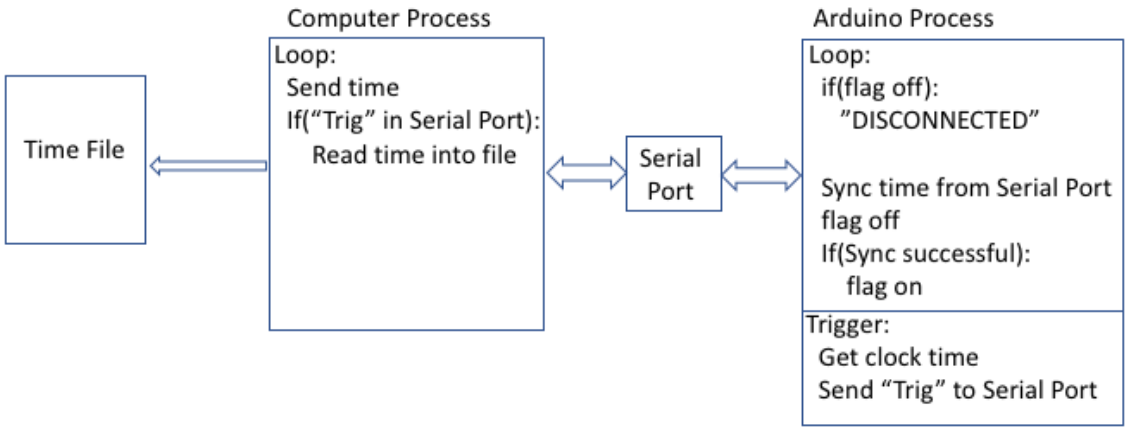

Figure 2: Software configuration for the setup. The communication is mostly one way unless the microcontroller is triggered.

Resource conflict

The reading and writing to the serial port must be carefully structured. Conflict in resource access can cause large delays to the synchronization process. If the computer and microcontroller are constantly competing to read and write to the serial port, it can be an extended time (on the order of 10s of seconds) before the microcontroller is able to read a time in from the serial port for synchronization.

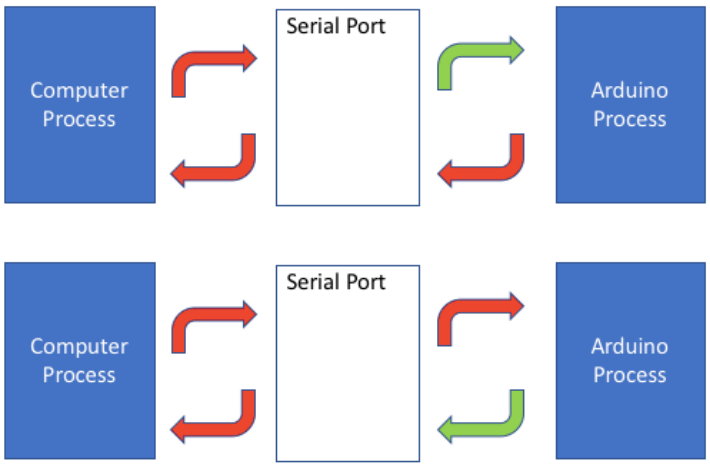

Figure 12: A typical issue dealing with spinlock. Access to the shared resource creates a conflict that can leave one process waiting on the other for extended periods of time.

To avoid this problem, I set up the two programs to read and write in one direction, with special care for the occasion when this flow is reversed. The computer process mostly writes to

the serial port while the microcontroller reads. Only when the hardware interrupt from the trigger occurs does the microcontroller send the time it recorded to the serial port, preceded with a special identifier. The computer process checks for a write from the microcontroller by looking for this identifier. When found, the computer reads the time and saves this to the time-stamp file. Importantly, the microcontroller sends the identifier and time during the interrupt service routine and then briefly delays its operation to allow the computer to retrieve this time before continuing operation.

Operation

To test the precision and the stability of this setup I connected the trigger to a function generator. I synchronized the function generator to a stable 10 MHz signal so that its output frequency could be considered exact. I sent regular triggers every 2 seconds and observed the resulting recorded times. I configured the microcontroller to send back the time to millisecond precision rather than the normal second precision. The results are shown in figure 13.

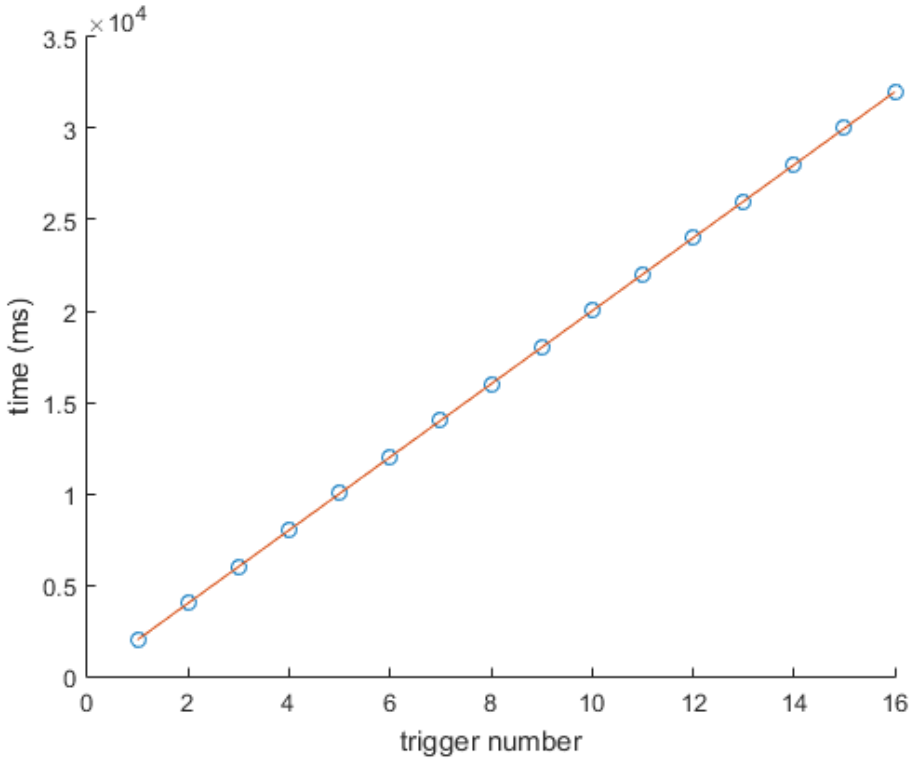

Figure 13: Precision and stability of clock over time. The time on the vertical axis is the time recorded at each of the evenly spaced triggers represented on the horizontal axis. The line represents ideal performance given a trigger every 2 seconds.

The line represents ideal performance for a trigger received every 2 seconds. The data fits this line well, with an average regular residual value of 1.75 milliseconds. Importantly, it does not drift away from the line over time, which would represent a concern for the stability of the clock. With this precision and synchronization to a time server, the microcontroller will always provide us time-stamps with the 1 second accuracy we need for our purposes.

Control of Instrument Settings

Motivation

Our experimental setup makes extended use of electronics. This includes a variety of instruments such as current controllers, function generators, and oscilloscopes. These represent a variety of manufacturers and models. Many of these instruments constantly operate at fixed settings.

Any experiment with use of a wide variety of electronic instruments faces an issue keeping track of the settings of each individual instrument. Failure to properly keep track of this can lead to a large amount of lost time in the face of events like power outages which can reset the instruments.

A solution to this problem requires some method of controlling these instrument settings and keeping track of what those settings should be. It should also be easy to use such that it saves significant time versus manually finding the desired settings and setting each of the instruments.

Solution

A simple solution that meets these requirements exists in the use of IEE-488 standard, commonly known as general purpose interface bus (GPIB). This standard can be used to set all of these instruments automatically with some simple software. Electronic instruments from a multitude of vendors like Tektronix, National Instruments, and Hewlett-Packard frequently have a GPIB interface. Many of the electronic instruments used in our apparatus provide such an interface.

GPIB supports parallel addressing of instruments. The connectors used can be attached to each other in addition to the instrument, so that multiple instruments can be attached in series. Each instrument has a unique identifier that the software uses to specify the recipient. This

makes it easy to connect all the instruments and then control them through a single software program.

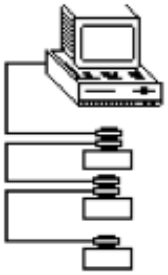

Figure 14: Example of parallel addressing in a "daisy-chain" configuration. Each connecter pairs with a device and another connecter, ultimately connecting all devices to a computer. The computer can send instructions to each device via an identifier.

Using MATLAB, I created a program to do this for many of our electronic instruments. MATLAB supports GPIB communication through its Instrument Control Toolbox, which makes it easy to create objects representing connected instruments and then send individualized instructions to each device.

The program I wrote is diagrammed in figure 14. The main program creates an object for each device based on GPIB addresses it finds in a file called address. It then runs a script called instructions which calls a series of scripts set*1*, set*2*, …, set*i* for each object *i*. These scripts simply contain SCPI commands to set desired settings for each particular instrument.

To add a device, all that is needed is to add the device's GPIB address to the bottom of the address file in a new line *n,* and add script set*n* to the directory containing desired SCPI instructions. This setup makes adding instruments to the configuration a simple task and simultaneously ensures that all settings are recorded and need not be looked up when devices are reset.

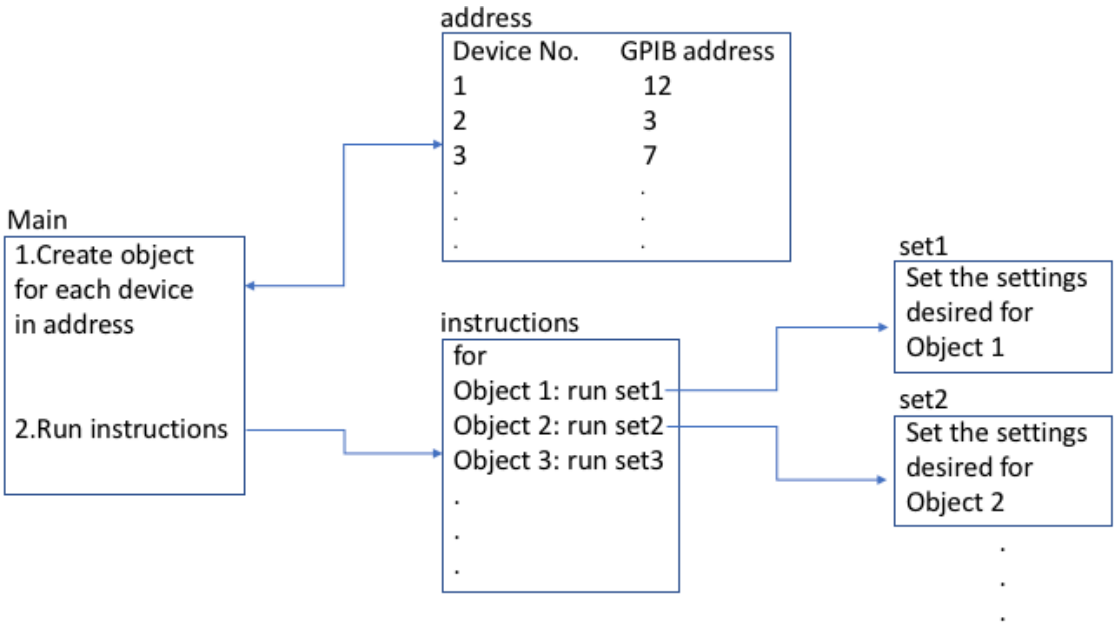

Figure 15: Software configuration for GPIB setup. The "main" program creates objects for instruments with their GPIB addresses defined in "address". It then uses the "instructions" script to send each object the specialized SCPI commands in a "set" script that corresponds to the object. Adding an object only requires adding it to address and adding its instructions to a new "set" script. Note that the GPIB address of the device must be unique. This can be set on the instrument as needed.

Conclusion

In summary, I made a number of enhancements to the apparatus. Using a magnetometer and analog-to-digital converter will provide an additional check on our magnetic field, which is critical to our experiment. This will remove data with background magnetic field noise beyond a threshold that can be set as low as 0.1 mG. It will also aid in debugging by tracking changes applied to the bias field inside the trap. On another front, the use of a microcontroller to keep time will always provide time-stamps accurate to 1 second and will thus improve data integrity in the experiment. Lastly, the software to control electronic instruments in our lab via the IEE-488 standard is an easy and convenient way to store instrument settings that will save time in the face of power outages.

Bibliography

[1] C. Chin, T. Grimm, P. Julienne, and E. Tiesinga. Feshbach Resonances in Ultracold Gases. *Rev. Mod. Phys.*, 82, 1225, 2013.

[2] B. S. Rem, A. T. Grier, I. Ferrier-Barbut, U. Eismann, T. Lan- gen, N. Navon, L. Khaykovich, F. Werner, D. S. Petrov, F. Chevy, et al. Lifetime of the Bose gas with resonant inter- actions. *Phys. Rev. Lett.*, 110(16):163202, 2013.

[3] R. J. Fletcher, A. L. Gaunt, N. Navon, R. P. Smith, and Z. Hadzibabic. Stability of a unitary Bose gas. *Phys. Rev. Lett.*, 111(12):125303, 2013.

[4] P. Makotyn, C. E. Klauss, D. L. Goldberger, E. A. Cornell, and D. S. Jin. Universal dynamics of a degenerate unitary Bose gas. *Nat. Phys.*, 10(2):116–119, 2014.

[5] U. Eismann, L. Khaykovich, S. Laurent, I. Ferrier-Barbut, B. S. Rem, A. T. Grier, M. Delehaye, F. Chevy, C. Salomon, L.-C. Ha, et al. Universal loss dynamics in a unitary Bose gas. *Phys. Rev. X*, 6(2):021025, 2016.

[6] R. J. Fletcher, R. Lopes, J. Man, N. Navon, R. P. Smith, M. W. Zwierlein, and Z. Hadzibabic. Two- and three-body contacts in the unitary Bose gas. *Science*, 355(6323):377–380, 2017.

[7] C. E. Klauss, X. Xie, C. Lopez-Abadia, J. P. D'Incao, Z. Hadzibabic, D. S. Jin, and E. A. Cornell. Observation of Efimov Molecules Created from a Resonantly Interacting Bose gas. *Phys. Rev. Lett.*, 119, 143401, 2017.

[8] G. Breit and I. I. Rabi. Measurement of Nuclear Spin. Phys. Rev. Lett., 38, 2082, 1931.

[9] J. Parthasarathy "Positioning and Navigation System Using GPS", *International Archives of the Photogrammetry, Remote Sensing and Information Science*, Volume XXXVI, Part 6, 2006

Appendix A-Feedback on microcontroller time

I connected the Arduino microcontroller to an LCD screen with the setup shown in figure 16 as a convenient way to provide additional feedback on clock synchronization. The microcontroller's time, whether synchronized or not, is displayed on the screen. If it is unsynchronized, and additional "DISCONNECTED" warning is displayed on the screen.

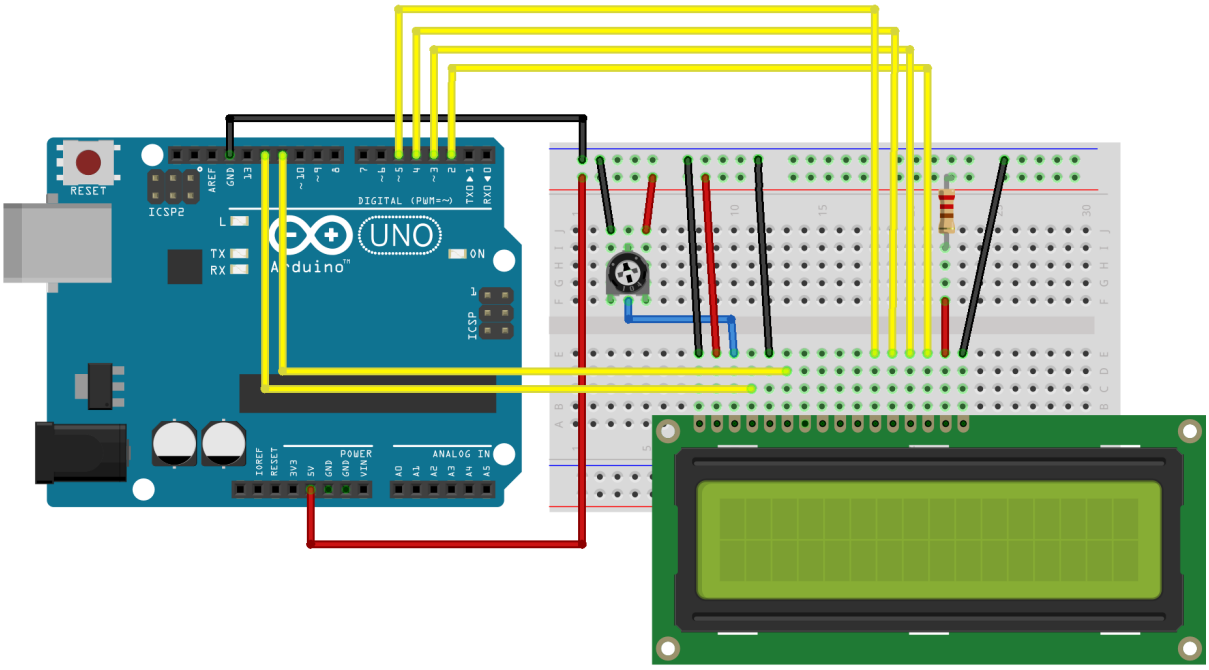

Figure 3: Diagram for wiring of Arduino UNO microcontroller to LCD screen. The resistor is 220 ohms.

The software setup is similar to the simplified version I already described. The difference is that between synchronizations, the microcontroller prints its current clock time to the LCD screen repeatedly, as well as a "DISCONNECTED" warning if a check for synchronization is not passed. Figure 17 shows a more detailed version of figure 11.

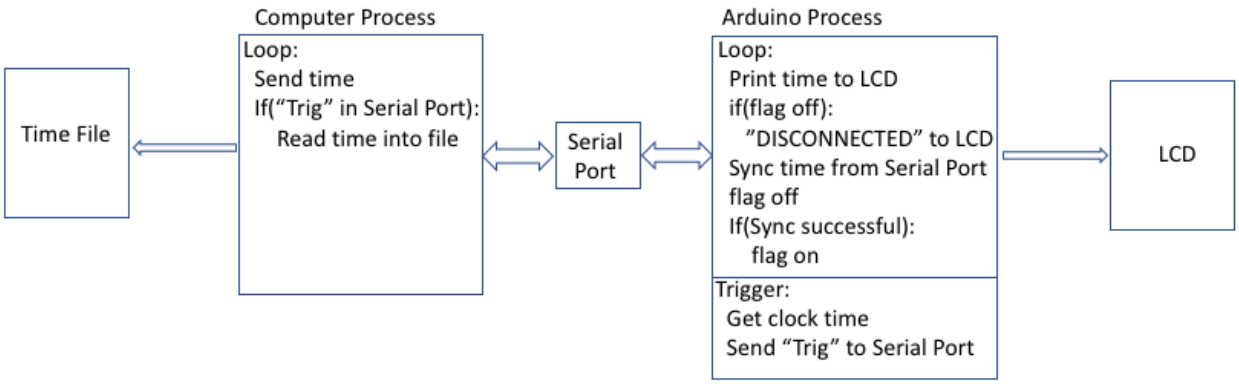

Figure 4: The software configuration of the microcontroller setup, including an LCD screen to display the microcontroller time and a warning if the microcontroller becomes disconnected.