An adaptive intensity stabilizer for optical lattices

Jakob Gillespie Hinney

Thesis submitted for the degree of

Master of Philosophy

of the University of Strathclyde

Supervisor:	Prof. Stefan Kuhr, University of Strathclyde
Internal Examiner:	Dr. Aidan Arnold, University of Strathclyde
External Examiner:	Dr. Matthew Jones, Durham University

Glasgow, August 2014

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Signed:

Date:

Abstract

The depth of a dipole trap for ultracold atoms is directly related to the intensity of the corresponding trapping laser beam. Therefore, accurate preparation and control of a trapped atomic cloud requires fine control over the intensity of the involved laser beams. This can be achieved with a controller that monitors the intensity on a photodiode and regulates it via an acousto-optical modulator, stabilizing it to a given value via a feedback loop. Here, two important benchmarks are the regulation bandwidth and the accuracy of the controller. Our experimental context requires fast and accurate intensity control of laser beams that form an optical lattice. An experimental sequence requires both intensities on a mW and on a W scale, where the controller performance at low intensities is particularly important as the physical systems of interest are studied in relatively shallow optical lattices.

This thesis presents an intensity controller that satisfies these requirements. To ensure good resolution in the low intensity regime, the controller employs a logarithmically amplified photodiode that is very sensitive in this regime. Since the gain of this photodiode varies drastically across different intensity regimes, it is impossible to optimize the gain of a static feedback loop for all intensities. Here, our solution is to dynamically adjust the gain of the controller using an Arduino microcontroller. We implement a gain schedule with five different gain regimes that optimizes the controller bandwidth across the entire intensity spectrum. For intensities below 10% of the total power we observe a substantial improvement, with a feedback bandwidth up to five times higher than in the case of a static PI-controller. The control accuracy at low intensities, i.e. how accurate a setpoint level can be specified, is at least 0.01%. The noise of the stabilization and thus the precision is better than 0.1%. The implementation of the Arduino allows for further controller configurations, enabling a wide range of applications such as completely digital PID control, storage of output values and prewarming of the AOM.

Acknowledgments

The past two years in the Photonics group were an extremely rewarding time for me. Starting as an undergraduate student with little lab experience I completed my first project in the group, only to come back as a Masters student one year later, working on the presented project. This experience would not have been possible without a great number of people that I would like to thank here.

Firstly, I am truly thankful Prof. Stefan Kuhr for the support and encouragement throughout the past year. He enabled me to continue my Physics education as a PhD student in a great group and a great place.

Aidan Arnold and Matthew Jones, for kindly agreeing to be the internal and external examiners of this thesis.

Elmar Haller, who supervised this work and gave his opinion and great ideas at all stages of the project.

I shared my office for the first half of the past year with our former postdoc Graham Bruce which lead to many great discussions and chats about almost everything. Graham gave me detailed feedback for presentations and this thesis and I am sure we will keep this contact in the future. Thank you.

The group members around the quantum gas microscope: James Hudson, for singing extremely strange songs with me in the lab. Dylan Cotta, who was a great gym buddy and also my flatmate in the past weeks. Andrew Kelly, who always took the time to give me feedback on a poster and this thesis. And Bruno Peaudecerf, who joined our time half a year ago and has been a good office fellow since then.

It was a great pleasure to spend time with the other members of the Photonics group. Our weekly football matches were always a highlight and our trip to the EGAS conference was quite memorable. I am impressed how much we grew together over the past year. Thank you, James McGilligan, Chris Carson, Pedro Gomes (Pedro Gomes, Pedro Gomes), Billy Robertson, Yueyang Zhai, Ivor Kresic and Savino Piccolomo. I also want to thank Paul Griffin for being such a supportive and and helpful member of the group.

In the past year our Department welcomed Andrew Daley as a new member and I learned much in the group discussions with him about lattice physics. With him came also Alexandre Tacla who I met in a Brazil jersey on the memorable day when his country conceded one of the worst defeats in the recent history of football (good sports spirit though!).

This project would have been impossible without the help of John Broadfoot and Ken Gibson from the electronics workshop. Thank you for the support and the entertaining time. It was a pleasure to work with you.

I am very thankful to my parents and my sister for the support of my studies and the interest in my life in Scotland.

Finally, thank you, Maria, for being my partner in crime throughout the past year in Dundee and Glasgow. I am excited for our future life in Vienna!

Contents

1. Introduction

Experiments with ultracold atoms are a powerful testbed for studying fundamental principles of quantum systems due to their weak environmental coupling and many tunable parameters. Since the first experimental realization of a Bose-Einstein condensate in 1995 [1, 2] this field has seen tremendous growth with research groups studying both bosonic and fermionic atomic gases in various trap geometries and interaction regimes [3, 4].

A prominent example is the case of ultracold atoms in optical lattices, arrays of dipole traps created by counter-propagating laser beams. It has been suggested that ultracold atoms trapped in these arrays closely resemble electrons in solid-state materials, where the optical lattice plays the role of a lattice of positively charged ions and ultracold atoms can exhibit the same mobility and interaction dynamics as electrons in these materials [5, 6]. While the physical mechanisms governing the evolution of both systems are almost equivalent, the experimental restrictions and manipulation methods differ substantially: Optical lattices are virtually defect free, the depth of the traps and thus the mobility of ultracold atoms in the lattice can be controlled accurately via the intensity of the constituent laser beams and even the interactions between the atoms can be manipulated via scattering resonances, so-called Feshbach resonances [7]. On the other hand, electrons in solid-state bodies cannot be probed and manipulated with equal ease: their electric repulsion cannot be modified, lattice mobility cannot be adjusted easily, thermal vibrations and impurities of the ionic crystal give rise to additional effects that superpose with the fundamental mechanisms of electron mobility. The different accesibilities for these analogous systems inspired the use of ultracold atoms in optical lattices for simulating electrons in a solid-state crystal lattice, a concept known as quantum simulation [8, 9].

The role of many-body states

The dynamics of electrons in solid-state materials remain a hot research topic to the present day, not least because astounding phenomena like high-temperature superconductivity still defy a complete theoretical description. Understanding the underlying principles of such quantum phases, however, might enable the design of new quantum materials with properties specifically tailored to a particular purpose. Therefore, great research efforts, both in the theoretical and experimental realm, have been directed at studying the physical principles of electronic systems. These seem to be strongly governed by interactions between many particles, making electrons in solid-state materials a complex many-body system whose emergent properties cannot be understood from its constituent elements alone. For example, quantum phases such as superconductivity seem to be entirely interaction induced, i.e. a single electron cannot become superconducting. The transition between such a superconducting phase with high electron mobility and an insulating phase with suppressed electron mobility therefore should depend crucially on the interplay between electron-electron interactions and other important energy scales.

The Hubbard model

The above considerations led physicist John Hubbard in 1963 to the development of a theoretical model that aimed at covering the mentioned quantum phases of electronic materials. In its original version, the Hamiltonian of this model includes only two terms: The first term describes hopping of electrons between ionic nodes of a crystal and the second term accounts for the electro-static repulsion between two electrons. More than thirty years later it was noted by the ultracold atoms community that the Hubbard model presents a powerful approach to describing ultracold bosonic atoms in an optical lattice, leading to the Bose-Hubbard model [10].

Following these theoretical developments, it was thus a milestone of quantum simulation when Greiner *et al.* reported on the observation of a quantum phase transition of ultracold Rubidium 87 in an optical lattice between a superconducting phase and an interaction induced insulating state, called a Mott insulator [11]. The researchers loaded a ^{87}Rb Bose-Einstein-Condensate (BEC) into an optical lattice, leading to a super-fluid phase where atoms were delocalized over many lattice sites and maintained their coherence properties. Subsequently, an increase of the lattice depth suppressed atom mobility

and enhanced the repulsive atom-atom interactions, triggering a quantum phase transition to a Mott insulator. This state state is characterized by strong localization of atoms to individual lattice sites, a well defined number of atoms per site, and a loss of atomic coherence.

Towards simulating electrons

The above experiments with bosonic systems were followed by an increasing interest to study fermionic systems that more accurately resemble fermionic electrons [12]. Here, the key lies in the Pauli exclusion principle, which states that no two fermions can occupy the same quantum state. Thus, the lowest vibrational state on a lattice site can only be occupied by one atom of each spin state, which in the case of cold atoms usually correspond to different Zeeman sublevels of the ground state. This principle is included in the Fermi-Hubbard model, a modified version of the Bose-Hubbard model. Here, a hallmark experiment was the observation of a quantum phase-transition of fermionic Potassium 40 between a metallic, conducting phase and a Mott-insulator [13] in 2008 by Jördens *et al.*.

1.1. The Quantum Gas Microscope

Unlike electrons in real materials, experiments with ultracold atoms provide direct detection methods. Collective properties of the quantum system such as temperature, momentum distribution and the spin states of the atoms can be inferred from time-of-flight (TOF) imaging and Stern-Gerlach spin separation. As for the former, the optical lattice and all other confining potentials are ramped down to let the cold atom cloud expand over the course of typically a few milliseconds before taking an absorption image, i.e. flashing on a resonant laser beam and recording the cloud's shadow. This technique thus provides information about the momentum distribution of an ultracold gas and properties such as matter wave coherence (Fig. 1.1a) and cloud temperature can be extracted. For fermionic systems, even Fermi surfaces can be observed when the lattice is ramped down adiabatically (Fig. 1.1b) [14], thus revealing the quasi-momentum distribution of the gas. Stern-Gerlach imaging relies on the different potential energies of magnetic sublevels in a magnetic gradient field. After switching off the optical lattice a strong magnetic field gradient is applied which leads to a hyperfine state dependent

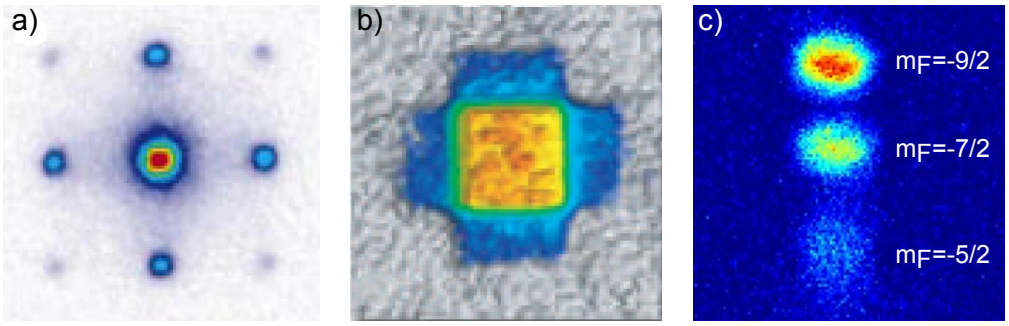

Figure 1.1.: a) Time-of-flight (TOF) absorption image of a BEC in an optical lattice: Coherence of the matter waves leads to well defined interference maxima. The image was obtained after suddenly switching of the optical lattice potential and taking an absorption image after 15 ms expansion time (Figure reproduced from [11]). b) Imaging the Fermi surface of a Fermi gas in an optical lattice: As opposed to a) the optical lattice is ramped to zero over the course of 1 ms followed by 9 ms of ballistic expansion. This maps the quasi-momentum distribution in the optical lattice onto the free-space momentum distribution, thus revealing the 2D Fermi surface (Figure reproduced from [14]). c) Stern-Gerlach imaging of ^{40}K: An atomic sample in the $|F = 9/2\rangle$ hyperfine ground state containing different m_F states expands in the presence of a strong magnetic gradient. This separates different m_F states due to different respective Zeeman energies (Figure produced in our experiment).

force. After a short evolution time an absorption image is recorded which shows a separate atom cloud for each magnetic sublevel (Fig. 1.1c).

Resolving single atoms

Spatial structures in the optical lattice cannot be resolved with these probes since the size of the expanded atom cloud is many times larger than the originally occupied optical lattice. Building on earlier work at detecting single atoms in large-period optical lattices using fluorescence imaging [15, 16], it was thus an important milestone when groups at Harvard university and the Max-Planck-Institute in Munich reported on in-situ imaging of individual atoms of ^{87}Rb in a Hubbard-regime optical lattice. An extremely high optical resolution close to 532 nm, the separation of single lattice sites, allowed for the spatial resolution of single lattice sites and led to the observation of spatial structures in a Mott-insulating state [17, 18] and the spatial distribution of entropy (Fig. 1.3a). In these experiments, the optical lattice potential is ramped up at the end of an an experimental

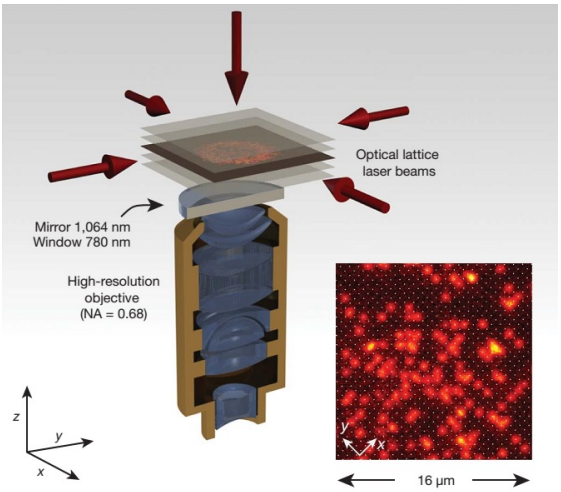

Figure 1.2.: Imaging the atom distribution of a single lattice plane: Fluorescence light is collected with a microscope objective and directed onto a camera to recorded the atom distribution (inset). Due to light assisted collisions, the atom distribution is observed modulo 2, i.e. doubly occupied lattice sites appear empty, three atoms on the same lattice site appear as one atom etc. (figure reproduced from [18])

sequence to confine the atoms in deep, isolated potential wells, thus freezing the spatial distribution. The atoms are then illuminated with near-resonant optical molasses light such that they fluoresce, while being cooled by the molasses at the same time. The fluorescence light is collected with a high resolution microscope objective and imaged onto a camera, as depicted in Fig. 1.2, revealing the atom distribution of a lattice plane with single-atom precision. Subsequently, the group in Munich also demonstrated manipulation of single atoms [19] (1.3b), paving the way to prepare quantum systems with high fidelity in order to study processes such as the mobility of spin impurities [20].

A microscope for fermions

The Quantum Gas Microscope at the University of Strathclyde aims to achieve this remarkable level of optical access and control for the fermionic species ^{40}K. Here, single-site resolved observation of a fermionic Mott-insulator is one important goal, following which the spatial entropy distribution of the fermionic Mott-insulator can be studied with unprecedented precision. Unlike the previously studied bosonic systems, ^{40}K atoms obey the Pauli exclusion principle. This gives rise to another insulating state, called the band

insulator. Here, in a fermionic gas of atoms in two magnetic spin states, corresponding to the two spin states of electrons in solid-state crystals, the lowest vibrational state of each lattice site is populated by two ^{40}K atoms. This phase resembles an insulating material without vacant electron states in the conducting electron band. Due to the well-defined number statistics and the Pauli exclusion principle, a defect-free band insulating phase can only be realized in one number-configuration, i.e. two atoms in the lowest vibrational state per lattice site, a state of extremely low entropy. It has been suggested that this state can be the starting point for an adiabatic transformation of the atom distribution towards a low-entropy Mott insulating phase (effectively reducing the occupation of each lattice site by one atom) where anti-ferromagnetic ordering can be observed, i.e. atoms on neighboring lattice sites have anti-parallel spin [21, 22]. Short-range magnetic correlations have been studied in a dimerized lattice systems where entropy was preferentially stored in weakly coupled links between lattice sites [23] but due to experimental restrictions the observation of long-range magnetic ordering remains an elusive goal[1]. Here, the ability of the Quantum Gas Microscope to resolve the spin distribution of a lattice plane provides a novel probe where one can directly obtain correlation functions between atoms separated by many lattice spacings.

Current stage of the experiment

At the current stage of the experiment we have successfully aligned the high-resolution objective and observed single ^{87}Rb atoms in the optical lattice (Fig. 1.3c) using a molasses cooling technique. For fermionic ^{40}K, however, we were not able to observe sufficient cooling with this technique, possibly due to a smaller hyperfine splitting of the excited state or due to harmful tunneling dynamics. Therefore, we currently implement a Raman-sideband cooling scheme [24] that makes atoms in the lattice fluoresce while simultaneously cooling them during the imaging phase, which has been demonstrated for ^{87}Rb [25, 26].

1.2. Intensity control

Experiments with single-site resolved detection methods require both shallow lattices for studying Fermi-Hubbard physics and deep lattices for the imaging process. An

[1] Just before submitting this work we became aware of a recent achievement by Hart *et al.* who report on the detection of antiferromagnetic correlations via Bragg scattering **??**

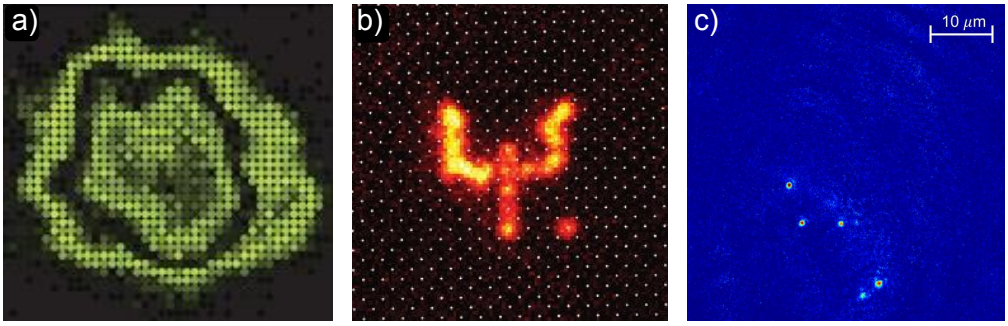

Figure 1.3.: a) Shells of a bosonic Mott Insulator: Due to light assisted binary collisions only lattice sites populated by odd atom numbers appear as occupied. The outer ring is a domain with one atom per lattice site, the adjacent inner dark ring with two atoms per lattice site, etc., since the chemical potential varies across the lattice (figure reproduced from [17]). b) Spin manipulation in a Mott insulator: A highly focused laser beam tunes atoms of only one lattice site into resonance with a microwave field based on the AC stark shift. After changing the hyperfine states of selected atoms in this manner the unaffected atoms are removed from the lattice with a resonant pulse and the remaining atoms are imaged (figure reproduced from [19]). c) Fluorescence image of single atoms at the University of Strathclyde: Five ^{87}Rb atoms are imaged in the lattice using molasses imaging. The atom with lowest luminosity was passibly lost during the imaging sequence and thus appears darker.

experimental sequence of the Quantum Gas Microscope typically consists of a first stage where atoms are loaded into the lattice and a system of interest is prepared and a second stage where the atom distribution of that state is frozen to allow for imaging which can last for up to a second.

The depth of an optical lattice depends on intensities of the laser beams that create the lattice. It is common to denote the lattice depth in recoil energies $E_r = (\hbar k)^2/2m$, corresponding to the kinetic energy of an atom with mass m and the momentum of a lattice photon $\hbar k$, where $k = 2\Pi/\lambda$ is the wavenumber of photon with wavelength λ. For instance, the wells of a 3D cubic 1064 nm optical lattice with a depth of $1E_r$ have trapping frequencies of $\omega/2\pi = 4$ kHz (^{87}Rb) and 9 kHz (^{40}K). The previously described lattice physics of quantum phase transitions mostly takes place in optical lattices of a depth between 0 and 30 E_r. For deeper lattices the atom mobility is almost completely suppressed and thus the quantum system bears little resemblance to the conducting or close-to-conducting materials that one aims to simulate. It is crucial to suppress atom tunneling during the imaging sequence, however, the interrogation of the atoms with

light almost inevitably causes heating. Thus, it is important to ensure that even heated atom samples are still confined strongly enough to preserve the initial lattice occupation. To achieve this the imaging sequence takes place in a lattice depth of hundreds or even thousands of E_r.

Requirements for an intensity stabilizer

The work presented in this thesis is the development of an intensity stabilizer to be used on the optical lattice laser beams. The above considerations outline the importance and requirements of an intensity stabilizer: the attainable resolution of the lattice depth should be below 0.1 E_r for shallow lattices but the regulation range has to extend up to thousands of E_r. Based on these considerations an intensity controller should be able to filter out fluctuations at frequencies on the order of the relevant trapping frequencies for shallow lattices and Raman-sideband-cooling. For the latter case, these can extend up to 100 kHz and the controller bandwidth should thus extend well beyond these frequencies in order to filter out noise at trapping frequency that would otherwise lead to heating of the atoms. The key requirements of a large dynamic range and a modulation bandwidth well above 100 kHz are addressed by two particular measures outlined in detail in Chap. 3. In short, the former requirement led us to implement a logarithmic photodiode that essentially magnifies the resolution of low laser intensities at the expense of a coarser resolution at high intensities, an acceptable trade-off. The latter resulted in the development of the presented PI controller as an analog circuit rather than a digital feedback loop. The non-linearities introduced by the logarithmic photodiode and the response curve of accousto-optic modulators require an adaptive PI tuning scheme which initially suggested the use of a high-speed digital PI controller. However, after initial tests with FPGA boards it became clear that a digital approach struggles to satisfy the bandwidth requirements due to the relatively slow analog-to-digital and digital-to-analog conversion which could be sped up only with substantial financial efforts. The presented controller is in fact a hybrid that consists of analog operational amplifiers for generating the feedback but uses an Arduino microcontroller to schedule a gain parameter to optimize the feedback for different lattice depths. Thus, we harness the advantages from both digital and analog controller, that is easy implementation of logic as required for gain scheduling and high bandwidth signal processing respectively.

Structure of the thesis

This thesis is structured as follows: In Chapter 2 we present an introduction to control theory and feedback systems, give an intuitive explanation of PID control and introduce control theory concepts such as adaptation, gain scheduling and hysteresis. Chapter 3 gives a thorough description of the presented intensity stabilizer, the experimental setup and the interaction between the analog circuit and the Arduino microcontroller. We also present evaluations of the stability and bandwidth of the controller to demonstrate that it complies with the mentioned requirements. Lastly, Chapter 4 places the controller in the context of the experiment and comments on its flexibility which renders it useful for other applications in the laboratory.

2. Control Theory

A controller is a device that regulates a parameter of a process to match a certain value based on a measurement of that value. Such a controller is necessary wherever one aims to adjust a process parameter that does not have a perfectly linear response to a process input signal. Consider an oven to be held at a particular temperature. If the temperature were a perfectly linear function of the currents through a heating wire, one would simply need to measure the temperatures at two different currents and could extrapolate the currents required for every other temperature. In reality, however, the oven temperature would rise to an unpredictable value and might overshoot before it loses as much heat to its environment as the wires provide. This suggests that the currents should be applied more carefully to achieve a desired temperature, but how exactly could it be done?

The principle of feedback

The above example illustrates why a great number of processes, both in industry and research, are regulated by controllers that solve the described problem. The fundamental concept is always the same and is depicted in Fig. 2.1: A process variable y has to be fixed to a certain setpoint y_{sp}. The means to adjust y are in the form of a process input parameter u, corresponding to the wire currents in the previous example. After application of the input u to the process the outcome is measured, i.e. y, and compared to the setpoint, that is computing the error $e = y_{\mathrm{sp}} - y$. Using this error to compute the next control parameter $u = u(e)$ is the essence of a feedback system and the described algorithm is the most basic description of a controller. Practical realizations of controllers come in various forms with the most common being the PID controller, or Proportional-Integral-Derivative controller. The appropriate type of controller depends on the particular way in which the process input u is obtained from the error and in the next section we discuss the most basic controller algorithm, known as On-Off control, and the widely used PID algorithm.

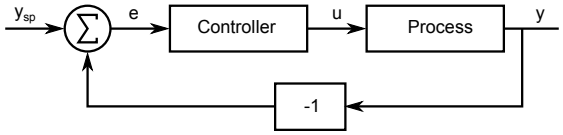

Figure 2.1.: A simple feedback loop: A parameter is measured and compared to a reference. The deviation is processed and fed back to the control input [27].

2.1. On-Off control

A simple approach to the described problem of heating an oven is the so-called On-Off-controller. Whenever the temperature is below the setpoint, the wire current is set to a constant value such that as soon as the temperature exceeds the setpoint the current is turned off. The output function $u(e)$ can be described by the following distribution

$$u(e) = \begin{cases} u_{\max}, & \text{if } e > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

However, we need to consider another important aspect of controllers known as the *dead time*. This refers to the time between applying u to the process and a resulting change in y. In the case of the oven it can take seconds or even minutes until the oven temperature changes due to altered wire currents and generally there will be a non-zero dead time for all real world processes. A large dead time means that an On-Off controller keeps the currents on longer than necessary. On the other hand, as the temperature subsequently drops below the setpoint it takes the dead time until the controller reaction stops it from dropping further. The result is temperature oscillations and the larger the dead time the more prone the controller to oscillations. Dead time is inherent not only in On-Off controllers but in all controller types.

2.2. PID control

The above oven problem illustrates why switching currents on and off it is not an effective controller algorithm. Instead, an algorithm that has proven to be extremely effective is the already mentioned PID-controller. Its advantage over the On-Off controller is that the output u is calculated in a more sophisticated way which accounts for system characteristics such as the dead time. Again, we illustrate this with the example of the

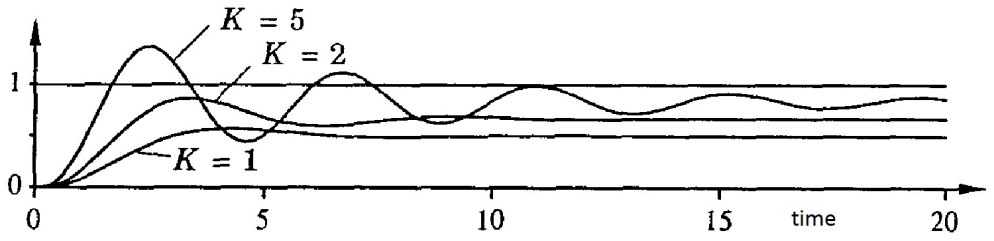

Figure 2.2.: Model of a proportional feedback controller: The controller cannot stabilize y at y_{sp} even if the gain is increased because the proportional feedback diminishes close to the setpoint. However, a larger gain makes the controller more prone to oscillations (figure reproduced from [27].

oven and the heating wires. At first instance, it might be useful to not only check if the control parameter y is below the setpoint but also how much it deviates. If y is only slightly below y_{sp} a small current is applied and if the deviation is larger the current is in a proportional way. An example algorithm is $u(e) = K_p e$, i.e. the output is proportional to the error. However, this alone cannot stabilize y at y_{sp} sufficiently. Consider the case where the control parameter is below the setpoint and one applies the error proportionally to the controller. This pushes the control parameter towards the setpoint, reducing the controller output towards zero. The result is that any environmental impact on the control parameter is able to push it away from the setpoint because the proportional feedback diminishes close to the setpoint. Thus, a proportional controller alone leads to a control parameter offset unless the setpoint matches the equilibrium controller position. If we try to counter this by increasing the gain factor K_p this will result in a smaller but nevertheless non-zero offset and the process variable will oscillate strongly in response to setpoint changes, as depicted in Fig. 2.2.

Integrating the error

This problem can be tackled by adding a component to the output that is not proportional to the error itself but to the area below the $e(t)$ curve, i.e. the integrated error. Indeed, in the above case the constant non-zero error would result in an increasing area below the $e(t)$ curve and thus to an increased output. This area, the integral of $e(t)$ over the time since the controller was activated, scaled by a constant K_i, converges only if the parameter y actually approaches y_{sp}, leading to a constant integral contribution just as $y = y_{\mathrm{sp}}$. Thus, when the proportional-integral controller has settled at the setpoint,

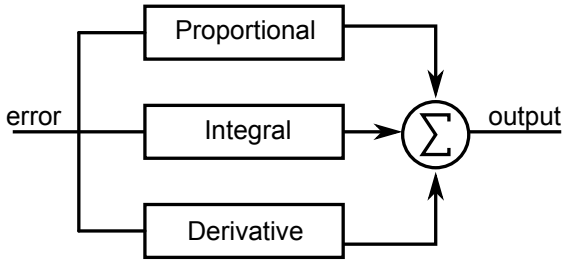

Figure 2.3.: The non-interacting PID algorithm [27]. All three components of the output are calculated in parallel and independent of each other as opposed to schemes where e.g. the output is calculated in consecutive steps.

the controller output is entirely given by the integral value and the proportional contribution is zero. Therefore, the integral gain has to be high enough to allow for integral output values corresponding to the entire range of desired setpoints. The optimum value of K_i depends on the controller dead time: If the integral 'charges' up substantially during the dead time it will subsequently push back the controller strongly, resulting in an overshoot and possibly even oscillations.

Derivative feedback

Adding an output component proportional to the integrated error can already stabilize y quite well if the proportionality constants K_p and K_i are chosen properly. Indeed, many controllers in industry and research use the PI algorithm. However, the following scenario suggests a third output component: Consider an external impact that suddenly changes the process parameter. In the first instance, the proportional feedback has little effect because the process parameter is still close to the setpoint and also the integrator component barely reacts because the area change below $e(t)$ is only non-zero since the disturbance and thus is not very large yet. One quantity that responds quickly to the disturbance is the change of the error $de(t)/dt = \dot{e}(t)$. Adding this derivative component to the output allows the controller to react when the error is only beginning to change but has not yet deviated much from zero, thus making it more responsive. On the other hand, a derivative feedback component can be very sensitive to noise in the process variable measurement and it might yield a more robust controller to employ only proportional and integral feedback. The exact choice here depends on the particular process and how it is controlled.

The PID algorithm

The three feedback components outlined are the core concept behind PID control. The output has the following form

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{2.2}$$

The performance of the PID algorithm is determined by the three tuning parameters, the proportional gain K_p, the integral gain K_i and the derivative gain K_d. Tuning these parameters to ideal values requires careful calibration, as will be discussed in the next section. The PID algorithm of Eq. 2.2 is depicted in Fig. 2.3 and is called the non-interacting PID controller because its three components are calculated in parallel and independent of each other [27]. Examples of interacting controllers include cases where the output is calculated in consecutive steps that pass on a transformed error signal to the next step. For this work we limit our discussion to the non-interacting form that corresponds to the developed controller presented in Chapter 3.

As previously mentioned, many real world controllers feature only PI feedback. Firstly, PI controllers may simply be sufficient for a given process. Secondly, derivative feedback is particularly sensitive to noise and can lead to noise-related oscillations. These reasons have led us to design the intensity stabilizer presented in this work as a PI controller.

2.3. PID tuning

Generally, a good controller stabilizes a process parameter as accurately and as quickly as possible. The latter point suggests increasing the introduced gain parameters to speed up the controller reaction to a setpoint change. However, if the gain is too large a small deviation from the setpoint pushes the process parameter back so strongly that it overshoots and starts to oscillate. Therefore, it is important to carefully determine ideal PID parameters. This can be achieved both via manual adjustment or analytic analysis of the system dynamics. As for the latter, two prominent techniques were proposed by Ziegler and Nichols in 1942 and involve the measurement of a controller response and the calculation of the parameters based on these measurements [28].

Ziegler-Nichols tuning schemes

The first technique is known as the *Step Response Method*: The response of the process variable to a sudden change of the process input is observed in an open-loop configuration, i.e. without feedback, and the dead time and the maximum slope of the control parameter are measured. Ziegler and Nichols presented a table from which one can look up PID parameters as a function of the dead time and the maximum slope. Secondly, the *Frequency Response Method* measures two properties known as the ultimate gain and the ultimate period. These are obtained by disabling the integral and derivative component of the feedback and increasing the proportional gain gradually to the ultimate gain, that is the point at which the control parameter oscillates with constant amplitude and frequency. Then one measures the period of this oscillation, called the ultimate period. Again, Ziegler and Nichols presented tables with ideal PID parameters as a function of the ultimate gain and the ultimate period.

Manual tuning

For this work, it has proven practical and straightforward to manually tune the PI parameters to optimal values rather than calculating them based on a tuning scheme. To do this, we increased the proportional gain below the onset of oscillations and the integral gain to the value where the closed-loop controller response to a step function lead to a small overshoot. This configuration is a trade-off of the inaccuracy due to an overshoot for an increased response speed of the controller.

2.4. Adaptive control

Static PI controllers, meaning that their tuning parameters are held constant, are challenged when the control process exhibits considerable non-linear behavior or the process dynamics are sporadically changing. In the case of non-linearities, the gain of the feedback loop itself depends on a process parameter which makes it difficult to choose optimal PID parameters. As discussed in the previous section, these parameters are chosen such that the controller reacts quickly but does not yet oscillate. Non-linearities, however, can lead to a situation where the gain is ideal for one range of the process variable but either too low or too high in another range. The former means that the controller response is slower and the latter leads to oscillations of y around the setpoint.

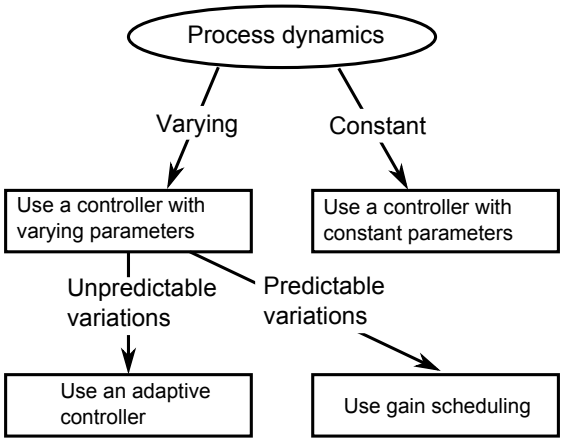

Figure 2.4.: A guide for choosing the right control technique depending on the process characteristics: While our intensity stabilizer exhibits varying process dynamics across the laser intensity range they are predictable and gain scheduling can thus remedy the gain adjustment problem [27].

To counter such problems the concept of adaptive PID controllers has been developed. One distinguishes between two approaches: Firstly, a controller is equipped to dynamically adjust its PID parameters to ideal values, commonly referred to *automatic tuning*. This is particularly useful if the gain of a feedback mechanism is not only varying but also unpredictable. Secondly, the controller chooses its parameters depending on a process variable, known as *gain scheduling*. This approach is preferable if the system dynamics are known and constant so that a table of optimal gain values can be defined from which the controller can look up its parameters. The flowchart in Fig. 2.4 is a guide for choosing among the different control techniques.

As will become clear in the next chapter, the scenario of a PI intensity stabilizer can include substantial non-linearities, however, these are constant in their behavior. Thus, the PI controller presented in this work features gain scheduling to achieve good responsiveness at all intensity levels.

2.5. Introducing hysteresis

Unlike the previously presented theory on controllers, real world circuits and controllers inevitably have an inherent level of noise which can harm the application of a gain schedule. Assume that a controller's gain is scheduled based on a process variable that

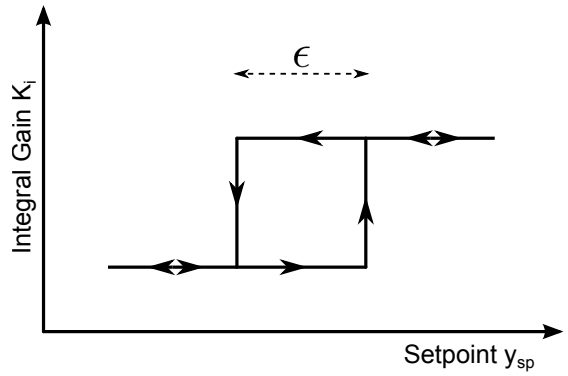

Figure 2.5.: The concept of hysteresis [27]: The threshold for changing the integral gain K_i depends on the direction in which the setpoint changes to avoid random gain flips due to noise when the setpoint is close to the boundary between two gain regimes.

is at a value close to the boundary of two different gain regimes. Small amounts of noise in the measurement of the process variable can now result in the controller flipping its gain randomly between the two regimes. Generally, changing the controller gain may take some time in which it cannot react to another, possibly more substantial change of the process parameter. Therefore, it might be beneficial to make the controller less prone to this using a concept called Hysteresis. This switching process is depicted in Fig. 2.5 and is also widely used for relay controllers [27]. The key idea is that the switching point depends on the direction in which the parameter is shifted. Consider again the gain scheduling scheme at a boundary voltage V_b between two gain regimes. With hysteresis the signal has to rise above $V_b + \epsilon$ for the gain regime to be flipped, rather than only V_b. In the same way, a decreasing signal has to drop below $V_b - \epsilon$ in order to cause a flip of the gain regime. If we choose the width of the hysteresis to be larger than the measurement noise we ensure that the gain is flipped only by actual shifts of the process parameter and not randomly due to noise. This concept will be demonstrated in the Arduino code presented in Sec. 3.3.2.

Summary

In this chapter we outlined the concepts of feedback control, including approaches to optimizing the feedback parameters and coping with varying system dynamics. These concepts should be considered when designing an appropriate controller for a given

process as the ideal choice depends on the process characteristics. In the next chapter we present a PI-controller for intensity stabilization of laser light that has to cope with strongly varying system dynamics due to a logarithmic measurement of the process parameter. The PI-controller automatically adjusts the gain of the integrator component to remedy the varying gain of the measurement device, based on the concept of Gain scheduling .

3. An adaptive PI controller

We present an intensity stabilizer for an optical lattice that complies with the two initially defined requirements: The controller has to provide high regulation bandwidth and accuracy while simultaneously covering a large range of intensities. Here, the particular focus is on the low intensity regime which corresponds to shallow optical lattices required for observing correlated quantum systems. To achieve good intensity resolution in this regime the controller employs a logarithmically amplified photodiode that is extremely sensitive for very low intensities. Since the gain of this logarithmic photodiode varies drastically across different intensity regimes it is difficult to optimize the gain of a classical PI feedback loop for this system. Our solution is to combine an analog PI controller with a digital potentiometer and an Arduino microprocessor that dynamically adjusts the gain of the feedback loop. This successfully remedies the problem of the varying photodiode gain and thus ensures that the controller shows good performance in all intensity regimes.

3.1. Experimental Setup

The optical lattice in our experiment is created by retro-reflection of three independent laser beams with wavelength 1064 nm. The two horizontal axes employ light from two Nufern fiber amplifiers that provide continuous wave output of up to 50 W, while the vertical lattice axis is provided by a 58 W master oscillator power amplifier (MOPA). The use of three seperate high-power laser systems is necessary to allow for sufficiently deep optical lattices during the imaging sequence. The light from the three lattice lasers is guided to the experiment through optical fibers and subsequently passed through an acousto-optical modulator (AOM) that deflects up to 90% of the light into an interference maximum depending on the amplitude of the radio-frequency (RF) signal applied to it. However, the fraction of light deflected is not a linear function of the RF amplitude, as can be seen from Fig. 3.1: At very low and very high intensities, the deflected proportion

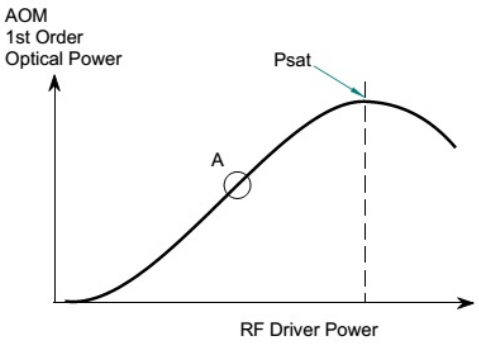

Figure 3.1.: Light diffraction in an AOM: The intensity diffracted into the first maximum depends non-linearly on the applied RF power and features a maximum beyond which the diffracted intensity decreases despite an increase in applied RF power (figure reproduced from [29])

grows more slowly than in the intermediate regime. The consequence is that the gain of an AOM depends on the intensity level, affecting the gain of any feedback loop that includes an AOM. Combined with the varying gain of the logarithmic photodiode (as described in the following section) these are the varying process dynamics mentioned in Sec. 2.4 that motivate the use of a gain schedule.

Directing the beams to the experiment

Since the spatial mode of the AOM-deflected beam can deviate notably from a Gaussian profile the deflected light is coupled to a single-mode optical fiber that filters out all non-Gaussian shape components and delivers a good beam profile to the optical setup around the science chamber[1]. The two horizontal lattice beams pass through the science chamber before being retro-reflected to create a standing light wave, whereas the vertical lattice beam enters the chamber through the top window and is retro-reflected from the bottom window of the science chamber. After the lattice beams have passed through the AOM, the fibre coupling and the optical setup, the maximum attainable power in the science chamber is around 16 W in each beam. To monitor their intensity we split off

[1]It is worth nothing that the deflection angle of the AOM can also vary with the applied RF power due to thermal expansion of the AOM's photonic crystal. This reduces the fibre coupling efficiency which strongly depends on the spatial alignment of the coupled beam. However, in our case the reduced efficiency is compensated by the intensity stabilizer. It has also been demonstrated, that applying a second RF frequency to the photonic crystal with a power such that it compensates the changes of the first RF frequency can reduce the thermal effects because the total RF power is kept contant [30].

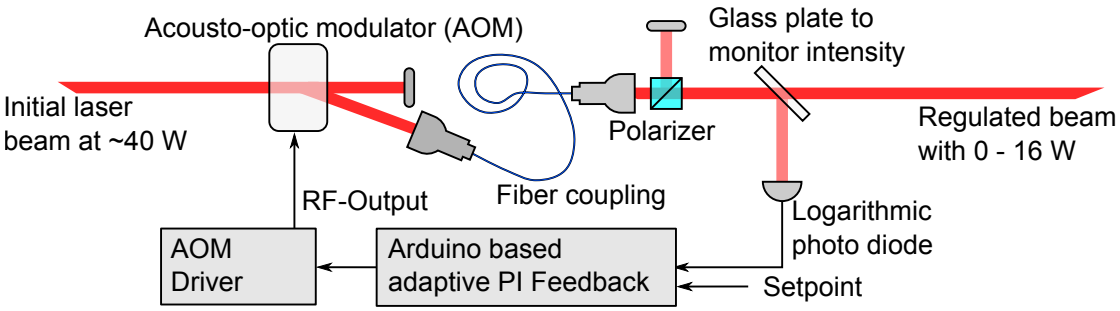

Figure 3.2.: Setup of the AOM based intensity stabilization. The feedback loop consists of the AOM driver, the AOM, the logarithmic photodiode and the adaptive PI-controller.

about 1% of the beam power via a glass plate and direct a part of it onto a logarithmic photodiode whose output signal is processed by the adaptive PI-controller presented in this work and detailed in Sec. 3.3. The controller output is connected to the AOM driver which applies an RF signal to the AOM, thus regulating the fraction of light that it deflects. A schematic of this feedback loop is shown in Fig. 3.1. The presented controller was tested with an optical test setup that resembles the described scheme. A picture of the test setup is shown in App. A.3.

Just as for every feedback loop, the bandwidth of the feedback is determined by the bandwidth of its constituent parts, i.e. the PI-controller, the AOM driver, the AOM and lastly the logarithmic photodiode. For the AOM driver and the AOM we measure an overall bandwidth of 3.5 MHz, well above the requirement of 100 kHz. Therefore, before examining if the intensity controller complies with the requirements we assess this for the logarithmic photodiode in the next section.

3.2. Logarithmic photodiode

The intensity controller contains a photodiode with logarithmic amplification to provide high sensitivity at very low light intensities. The corresponding circuit has been developed at the Max-Planck-Institute in Garching, Germany, and is centered around a transistor-based logarithmic amplifier (Analog Devices, `AD8304`). A schematic of the circuit is presented in App. A.1. The amplifier can be configured according to the specific requirements, as detailed below.

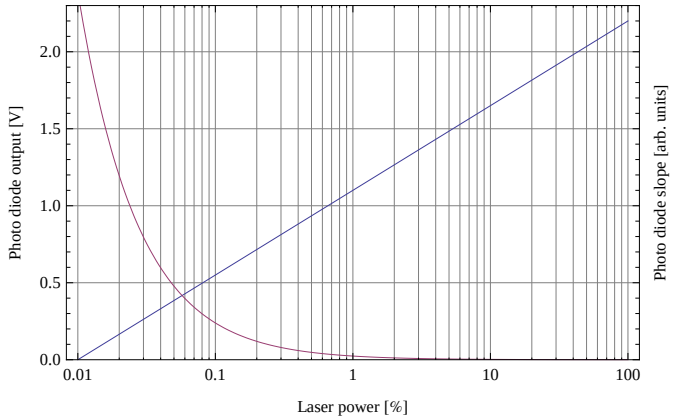

Figure 3.3.: The output voltage (blue) and slope of the logarithmic photodiode (red). The latter is equivalent to the photodiode gain and illustrates how it varies with intensity, resulting in an intensity dependent gain of the intensity stabilizer feedback loop.

3.2.1. Logarithmic amplification

The AD8304 makes use of the logarithmic relation between a bipolar transistor's base-emitter voltage and its collector current. Since this voltage and its zero intercept are both temperature dependent the AD8304 features an internal temperature compensation circuit such that both properties can be set to stable values via three periphery resistors. Here, we choose a resistor combination that results in a zero intercept of 500 nA photodiode current and a slope of 0.5 V/10 dB which we experimentally measure to be 0.52 $\pm$0.02 V/10 dB. In this configuration, the maximum output voltage of the photodiode is 2.2 V and for currents below the zero intercept the output is 0 V.

The chosen configuration of the logarithmic amplifier means that the photodiode can monitor light over more than four orders of magnitude. In our setup we couple out $\sim$1% of the lattice beams for regulation and monitoring with a glass plate, part of which we direct onto the logarithmic photodiode. Here, we adjust the incident power such that the maximum lattice power generates the maximum photodiode output voltage of 2.2 V. A plot of the photodiode voltage as a function of the scaled laser intensity is given in Fig. 3.3 together with a plot of the photodiode slope. Here, both the advantage and the challenge of the logarithmic amplification is illustrated. Regarding the former, consider an output voltage of 0.55 V, corresponding to 0.1% of the full power. The slope at this point amounts to 2.4 V/% while the slope at 10% has decreased by exactly two orders of

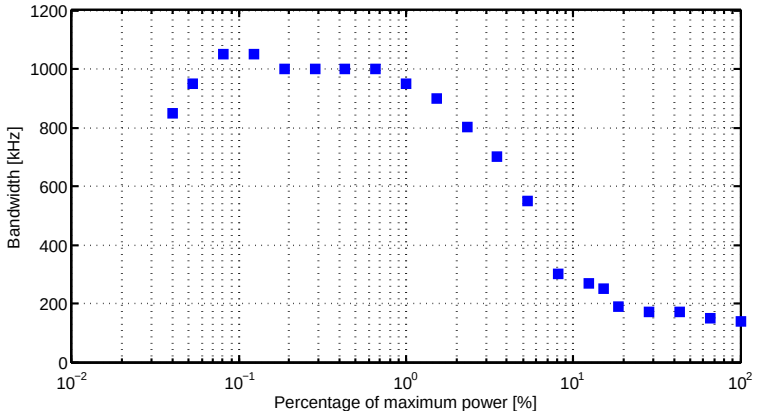

Figure 3.4.: Bandwidth of the logarithmic photodiode: For low intensities between 0 and 5 % it is at least 500 kHz, dropping to 140 kHz in the high intensity regime. Due to stray light from the AOM the minimum photodiode voltage was 0.33 V, corresponding to 0.04 %, so that bandwidths below that value could not be measured.

magnitude to 0.024 V/%. Thus, the logarithmic amplification of the photodiode current allows us to trade sensitivity at high intensities for sensitivity at low intensities. As the accuracy for shallow lattices is more crucial than for the deeper imaging lattice this is beneficial. On the other hand, this varying sensitivity has the consequence that the gain of the photodiode and thus of the feedback loop also vary with the intensity level. This is precisely the purpose of a gain schedule in our controller. To remedy the decreasing photodiode gain we increase the controller gain with gain scheduling as detailed in the next section. However, first we ensure that the logarithmic photodiode is suitable for an intensity stabilizer not only regarding its accuracy but also regarding its bandwidth over the whole intensity regime, as presented in the next section.

3.2.2. Bandwidth

As discussed in Chapter 2 the effectiveness of a feedback loop depends on the time that passes between the application of a controller signal and the measuring of the outcome. The larger this time difference is the more difficult is it to stabilize a process variable and in particular apply setpoint changes. Thus, a photodiode used for our intensity stabilizer has to provide a regulation bandwidth well above the bandwidth requirements for the final controller.

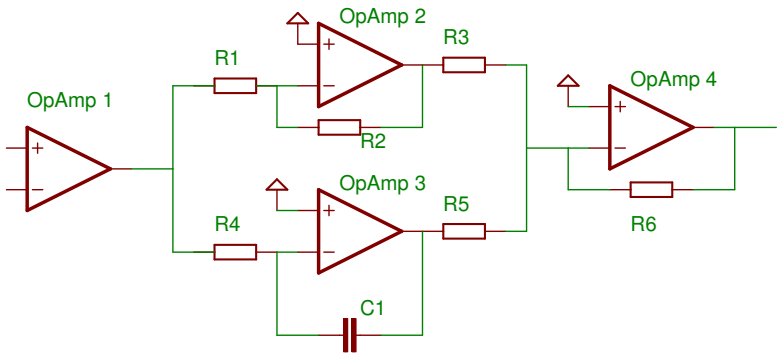

Figure 3.5.: The basic structure of an analog PI controller. `OpAmp1` generates an error signal which is processed by an op-amp with proportional amplification (`OpAmp2` and an integrating op-amp (`OpAmp3`). `OpAmp4` acts as a summing amplifier to generate the PI output.

To ensure this we evaluate the bandwidth of the logarithmic photodiode which we define as the frequency at which the monitored amplitude of a small[2] sinusoidal intensity modulation is decreased to 50%. The result is shown in Fig. 3.4. For low intensities, i.e. the output range from 0 to 1 V corresponding to 0 to 2% of the maximum beam power, the bandwidth is consistently above 800 kHz. Beyond this range the bandwidth drops to around 500 kHz at 1.5 V (∼5%) and 140 kHz at the maximum intensity. These values are in good agreement with the datasheet of the logarithmic amplifier `AD8304` [31]. The low intensity bandwidth is perfectly suitable for our controller requirements of 100 kHz regulation bandwidth. The drop in bandwidth for high intensities is acceptable for the same reason as in the case of the decreased intensity resolution in that regime. The deep lattices for imaging do not have to be as stable and accurate as the shallower lattices where the physical systems have to be prepared with high fidelity.

With the experimental setup outlined and an assessment of the monitoring photodiode we can now turn to the adaptive PI controller itself and describe how the interplay between a micro-controller and an analog circuit enables gain scheduling to encounter the varying system dynamics.

[2]While amplitude of the modulation was below 10 % for all measurements it was gradually increased with growing intensities to produce a photodiode signal with good signal to noise ratio.

3.3. Adaptive PI control

Before turning to the details of the analog control circuit we start with considering the most basic analog PI feedback circuit, as depicted in Fig. 3.5. It translates the PI algorithm described in Sec. 2.2 into an analog circuit based on operational amplifiers (op-amps). The two input signals from the photodiode and the experiment control system are sent to a unity-gain differential amplifier (`OpAmp1`) that outputs a voltage equal to the difference between the two input signals. This generates the error signal $e(t)$ which can now be processed in parallel by both the proportional amplification op-amp (`OpAmp2`) and an integrating op-amp (`OpAmp3`). The former is realized through negative feedback of the output via a resistor while the latter results from negative feedback via a capacitor. The output of the two Op-Amps is inverted and is given by

$$U_{\mathrm{Prop}}(t) = -\frac{\mathtt{R2}}{\mathtt{R1}} e(t) \tag{3.1}$$

$$U_{\mathrm{Int}}(t) = -\frac{1}{\mathtt{R4\ C1}} \int_0^t e(t')dt' \tag{3.2}$$

The fourth Op-Amp (`OpAmp4`) acts as a summer, adding up the inverted proportional and integral component according to

$$U_{\mathrm{PI}}(t) = -\mathtt{R6}\left(\frac{1}{\mathtt{R5}} U_{\mathrm{Int}}(t) + \frac{1}{\mathtt{R3}} U_{\mathrm{Prop}}(t)\right) \tag{3.3}$$

The output voltage $U_{\mathrm{PI}}(t)$ is then processed by an AOM driver to steer the amount of light deflected by the AOM.

Tuning parameters

The feedback of this basic circuit is governed by the circuit components mentioned in Eq. 3.1, 3.2 and 3.3. For this work, we will refer to `R2/R1` as the proportional gain, `1/R4C1` as the integral gain and `R6` as the loop gain. From the above equations it is clear that the various gain parameters of this PI controller all depend on the values of resistors in the circuit. Therefore, altering the resistances in the circuit alters the gain of the controller. This is the core idea behind the presented intensity stabilizer. In short, we use an Arduino micro-controller to adjust `R4` between 0 and 10 kΩ depending on the setpoint voltage, thus scheduling the integral gain of the controller. Adjusting the integral gain via a digital potentiometer proved to be the most noise free scheme compared to one

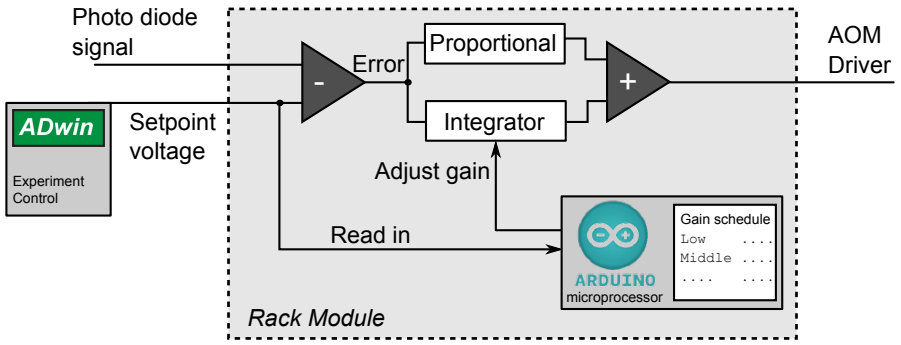

Figure 3.6.: Schematic of the Arduino based PI-controller with Gain scheduling. The Arduino continuously reads the setpoint voltage and compares it to the Gain schedule in order to adjust the integral gain.

where the proportional gain or the loop gain as adjusted. A schematic of this interplay is shown in Fig. 3.6.

A circuit board was designed to translate the described Arduino based PI feedback loop into a practical device for use in the lab, including protections and extensions to allow for flexible use of the controller, as detailed in the next section.

3.3.1. The circuit board

Developing the described basic PI controller circuit into a practical and robust device requires periphery circuitry as detailed in this section. The full Eagle schematic of the controller is shown in Fig. 3.7. At its heart is the previously described PI circuit where `IC3`, `IC4A`, `IC4B` and `IC4C` play the roles of `OpAmp1`, `2`, `3` and `4` respectively. The key feature, the variable resistor allowing for changing the gain, is embodied in `U1`, a digital potentiometer that can be adjusted via a digital interface. In the next paragraphs, we will go through most of the components of the circuit and explain their purpose.

(i) *Digital potentiometer for variable integral gain:* Adjusting the integrator gain is achieved via the `MCP42010` two-channel variable resistor that can tune the resistance between two pins between 70 and 10 kΩ[3]. This resistance, in series with a 200 Ω static resistor (`R34`), determines the gain of the integrator Op-Amp. Here, the negative feedback via a 2.2 nF capacitor (`C12`) leads to an integration constant $1/RC = 1/$`C12 (U1+R34)` ranging from more than 4×10^4 to 1.7×10^6 V/s, i.e.

[3]The minimum resistance is given by the inherent wiper resistance

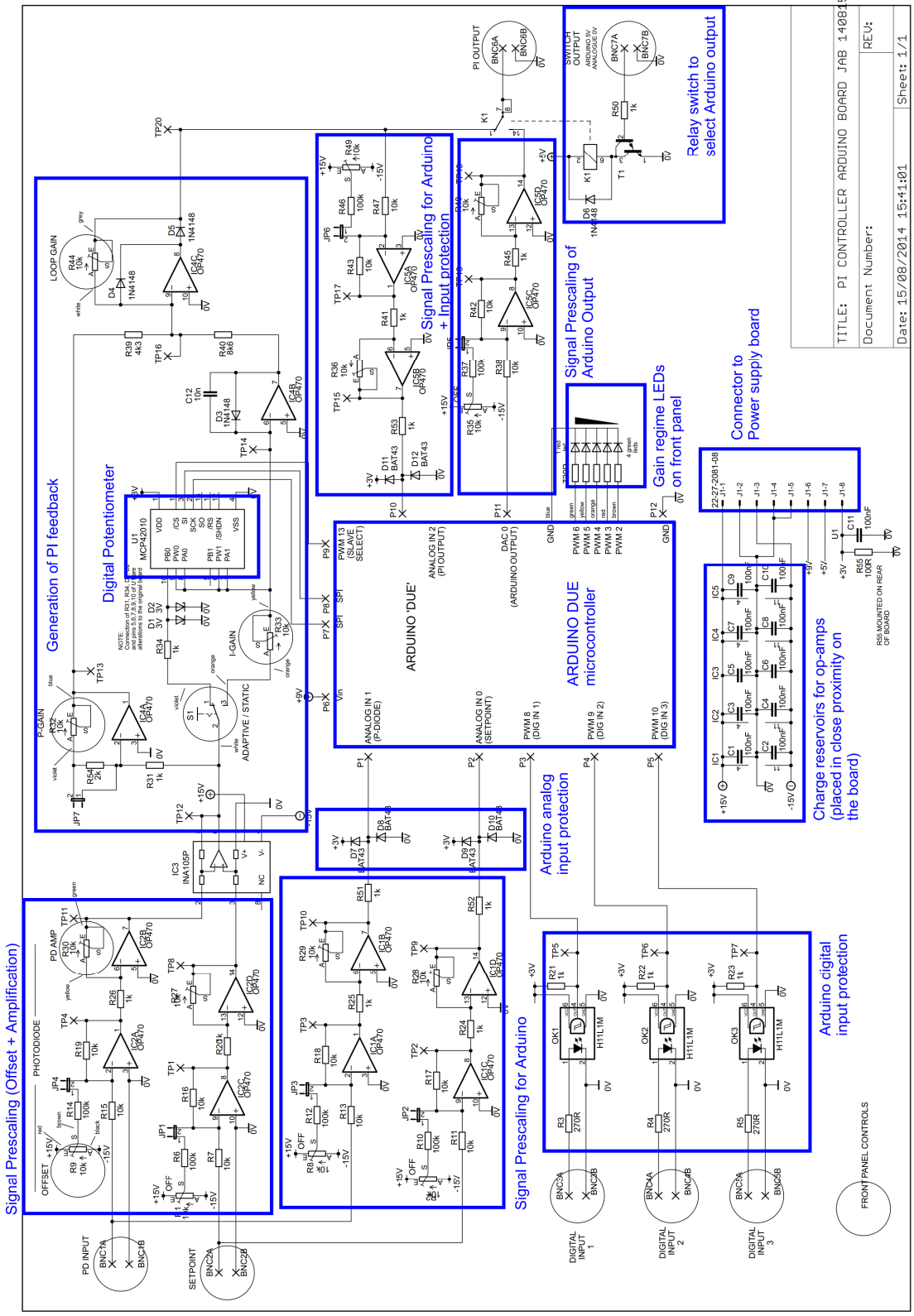

Figure 3.7.: Circuit schematic of the Arduino based PI-controller. The board is powered by a separate board shown in App. A.2.

the gain can be tuned over more than two orders of magnitude. This range of values has to correspond to the inertia of the controlled process, for example the gain would be too high for slower processes such as a temperature stabilization. To ensure a high flexibility of the the PI controller, the feedback capacitor is pluggable and can be changed easily. Additionally, `R34` can be resoldered easily as well to provide maximum flexibility of these gain parameters.

Communication between the digital potentiometer and the Arduino takes place via a Serial Peripheral Interface (SPI) that is a common standard for communication between a Master (here, the Arduino) and a Slave device (here, the `MCP42010`). In the case of one way communication from the master to the slave, only three wire connections are required: The Chip Select bus (`/CS` or pin 1) is at logic 1 while the slave device is operating independently and is flipped to logic 0 to select the slave device for data transmission. Once the slave is selected, the Arduino sends two 8-bit bytes to the chip by sending 16 clock cycles on the `CLK` bus and parallel to that the two bytes on the `MOSI` (Master Out Slave In) bus[4]. The first byte specifies the resistor channel of the potentiometer and the second byte the resistance between the pins 5 and 6 with a resolution of 10 kΩ/256 = 39 Ω. After transmission of the two bytes the chip select pin is restored to logic 1 and the potentiometer thus deselected. The chip select pin can be any of the digital Arduino outputs while the SPI signals `CLK` and `MOSI` are provided by specific SPI pins on the Arduino Due board. All connections required for SPI communication are achieved via wires from the controller board to the respective pins on the Arduino board which is mounted on the controller board.

The `MCP42010` is powered with V_{DD} = 3 V since the Arduino Due logic output ranges from 0 to 3.3 V and the potentiometer accepts a logic 1 only above 0.7 V_{DD}, e.g. 3.3 V would not be read as logic 1 by the potentiometer if it were powered with 5 V. However, the potentiometer cannot sustain potential differences of more than V_{DD} across the resistor pins and indeed it was found that the potentiometer breaks easily when this voltage is exceeded. Therefore, two protection Zener diodes are located between the error signal Op-Amp and the digital potentiometer to confine the voltage in this section between $\pm$3.3 V (`D1` and `D2`).

[4]The pin mapping of the Arduino Due can be looked up on the Arduino website under `http://arduino.cc/en/Hacking/PinMappingSAM3X`

For cases where the adjustable gain offered by the digital potentiometer is not necessary a manually adjustable potentiometer (R33) is placed in parallel, which allows the user to switch between the two configurations via a front panel switch (S1). This also allows us to compare the adaptive feedback loop to static PI loop.

(ii) *Prescaling the input signals:* Before comparing the photodiode input with the setpoint input signal, both signals are scaled by the amplification op-amps IC2B and IC2D and an offset may be added through IC2A and IC2C. The former is necessary since the logarithmic photodiode output ranges from 0 to 2.2 V while the experiment control output can extend up to 10 V. To make use of the full 18 bit resolution of the experiment control system, we tune the maximum photodiode signal to be equal to the maximum setpoint value. This is done by adjusting the feedback resistor R30. If the minimum photodiode voltage deviates notably from 0 V, for example due to stray light, it can be offset to that value by adjusting R9. If not needed, the offset can be completely deactivated by removing the corresponding jumper (JP4).

The same prescaling options are also available for the setpoint input. For the case of the intensity stabilization this might not be necessary but to ensure a high application flexibility of the circuit they were included regardless. If not needed, the offset jumper JP1 is removed and the gain of IC2D is set to 1.

The prescaled signals are then compared against each other by the unity-gain differential amplifier INA105 (IC3) to be processed by the PI circuitry.

(iii) *Rectifying the output:* The controller output is passed on to an AOM driver where the signal is mixed with an RF signal, such that the RF amplitude is proportional to the amplitude of the controller output. However, this mechanism is insensitive to the sign of the controller voltage with the consequence that at the lowest levels of controller outputs the circuit effectively regulates around an extremum. However, for negative controller output signals the feedback sign is reversed: The circuit reacts to a rise in the control variable by reducing the output signal, increasing the control variable even further. To prevent this, the output of the controller has to be rectified, i.e. the output cannot fall below 0 V. This is achieved with a precision rectifier added to IC4C in the form of the two diodes D4 and D5.

Another rectifying diode is included in the feedback of the integrator op-amp (D3)

that prevents the integral from charging up with negative values. This could otherwise easily be the case if the photodiode signal cannot fall down to the setpoint value, for example due to stray light from other beams. This rectifier configuration suffers from the diode-typical conduction threshold of ∼0.7 V which was circumvented by the two-diode-configuration described above. Here, however, this threshold is affordable since the purpose of the integrator rectifier is merely to avoid a substantial negative charging of the integral which decreases the integrator reaction time.

(iv) *Reading the input and output signals with the Arduino:* Gain scheduling requires monitoring of a process parameter in order to execute a gain schedule. The developed circuit enables the Arduino to read in both the photodiode signal and the setpoint. However, since the Arduino is operating with voltages not exceeding 3.3 V it cannot process signals above that voltage and both signals have to be scaled appropriately. This is done via the op-amps `IC2A, B, C`, and `D` as described in the previous paragraph on prescaling the input signals. To protect the Arduino from voltages above 3.3 V, diodes are placed at the Arduino input pins that will open if the voltages are outside the range from 0 to 3 V (`D7, 8, 9, 10`) and thus confine the voltage to this range. This has to be considered when tuning the analog inputs of the Arduino, i.e. the maximum input signals should be scaled to 3 V.

It can be very useful to read in the analog PI controller output with the Arduino, for example in order to store certain output voltages corresponding to a given setpoint value. Therefore, we also send the controller output to an Arduino analog input, again using two op-amps (`IC5A, B`) for scaling and offset to map the controller output range of 0 to 6 V (matching the AOM driver input range) onto 0 to 3 V. Again, additional protection is given by two diodes (`D11, 12`) confining the input voltages as described above.

(v) *Digital control inputs to the Arduino:* To control the Arduino during operation we included three digital inputs to the microprocessor in the circuit. Again, protection measures ensure that the Arduino input voltages do not exceed 3 V. Here, the problem is simplified by the fact that only digital signals have to be processed. Therefore, we use digital optocouplers with logic 1 output of $V_{\mathrm{CC}} = 3\ V$ (`OK1, 2, 3`).

(vi) *Analog output from the Arduino:* It can be useful and generally good for flexible operation to switch the controller output between the analog PI output and a signal generated by the Arduino. For the latter, we use one of the two 12-bit digital to analog converters (DAC) housed on the Arduino Due. However, once again these values are confined to the operation voltages of the Arduino. In fact, the DAC output does not fall below 0.04 V. Therefore, the output voltage has to be both amplified and offset to allow for an output range from 0 to 6 V, i.e. the input range of our AOM drivers. As before, the scaling and offset is achieved with two op-amps (`IC5C, D`).

To allow for switching between analog PI and Arduino output we have included a solid-state relay that can be operated through the experiment control system (Hamlin HE721CO510, `K1`). Applying 5 V to the corresponding front panel BNC connector switches from analog to Arduino output.

(vii) *Power supply:* The developed circuit is powered by a separate circuit board, providing 3, 5, 9 and ±15 V. The power supply board transforms the AC mains into DC ±18 V, from which voltage regulators generate ±15 V (Fairchild `LM7815, LM7915`). The +15 V output is processed further by voltage regulators to generate 9, 5 and 3 V in subsequent steps (`LM317, LM7805, LM317` respectively). The 9 V are required to power the Arduino Due, 5 V power the solid-state relay `K1` and 3 V are required as a power supply for the opto-couplers, the digital potentiometer and the Arduino diode protections mentioned above. All voltages connected are to ground via several capacitors to filter high frequency noise. More 10 nF capacitors are placed on the PI-controller board close to the various op-amps. These serve as a charge reservoir in cases where the op-amps quickly have to output currents and thus increase the response time.

The power supply board is connected to the PI controller board via an 8-pin connector; both boards are are housed in the same rack module.

We presented the developed circuit and described the SPI communication between the Arduino and the digital potentiometer. Next we turn to the Arduino microprocessor itself and the code it executes to explain how it dynamically adjusts the gain of the feedback loop.

3.3.2. The Arduino code

It was described that a core concept behind the adaptive intensity stabilizer is a predefined gain schedule, carried out by an Arduino Due microprocessor, that adjusts the integrator gain depending on the setpoint. In this work, we use a gain schedule that differentiates between five gain regimes. The Arduino reads in the setpoint voltage and checks to which gain regime, defined with an upper and lower voltage boundary, the value corresponds. Intuitively, this could look like an array of if-loops such as 'if the photodiode signal is larger than a and smaller than b then set the resistance to y'. The Arduino would then cycle through these if-loops infinitely and schedule the gain.

Optimizing the sampling time

However, this approach is not very efficient because for a constant photodiode signal the Arduino will apply the same resistance via the SPI functions in every loop cycle which unnecessarily increases the loop time. Every loop cycle has to begin with the Arduino reading the photodiode signal and if the overall loop time increases it also increases the sampling time of the photodiode signal. But the sampling time inevitably sets an upper limit for the responsiveness of the gain adaptation, i.e. how quickly the Arduino can detect that a different gain value is required. Therefore, it is desirable to keep the loop time as small as possible. We measure a sampling time of 42.7 μs for the configuration that updates the resistance every loop cycle. To improve this we implement the gain schedule such that the resistance is set not every loop cycle but only when the photodiode signal has shifted from one gain regime to another. To achieve this, we introduce a parameter in the gain algorithm that stores the value of the current gain regime. Now we can achieve the execution of the gain schedule with a slightly modified version of the if-loop array such as 'if the photodiode signal is larger than a and smaller than b *and the gain regime parameter does not correspond to this gain regime* then set the resistance to y'. The benefit is that the loop is rejected for constant photodiode voltages, decreasing the overall loop time and thus the sampling time to 9.8 μs, a more than fourfold improvement.

Introducing Hysteresis

Before we illustrate this concept with a real code snippet, however, we need to consider another practicality of the intensity stabilizer in order to cope with noise. Even for

constant photodiode signals the Arduino might read in different values both due to electric noise in the stabilization circuit and the inherent inaccuracy of the analog to digital converter on the Arduino board of around 1 bit. To make sure that the gain value is not flipped randomly when the setpoint is constant around the boundary of a gain regime we implement hysteresis (as discussed in Sec 2.5) into the algorithm by defining a 'dead zone' between gain regimes. For a decreasing setpoint value it has to fall below the lower end of the dead zone while for an increasing value it has to exceed the upper value of the dead zone.

Structure of an Arduino program

An Arduino program has a generic structure consisting of two parts, the initialization function `void setup()`, which is executed once when the Arduino is powered up, and the main code, written in the function `void loop()`, which is executed in an infinite loop following the initialization. Variables such as the gain schedule are defined outside both functions. A program is written to the Arduino via a USB connection and the Arduino software, shown in Fig. 3.8.

For our gain scheduling code, the main purpose of the `setup()` function is to initialize the SPI functions required for communicating with the digital potentiometer and to prepare the digital output pins for operation. The complete initialization function of our Arduino program is given in App. B.

The `setup()` function is executed only once when the Arduino is powered up or reprogrammed. Having done this, the Arduino executes the code in the `loop()` function until power is shut down or it is reprogrammed. The loop cycle starts with reading the photodiode signal and subsequently checks based on that value and the gain regime parameter if the signal has shifted to another gain regime via a sequence of if-loops. An illustrating excerpt of the corresponding Arduino code of such an if-loop is given below. The variable `measured` is the setpoint signal read by the Arduino. The code snippet tests if the setpoint has shifted from the lowest region to the second region. If this is the case, it flips the controller gain (line 3) by calling the function `digitalPotWrite(int CommandByte, int value)` that adjusts the resistance. It adjusts the region parameter required for the if-loop array and it turns off and on the corresponding LEDs on the front panel to indicate the correct gain regime.

```
if (measured > FirstRegionUpper && RegionParam == 1) {                    1
```

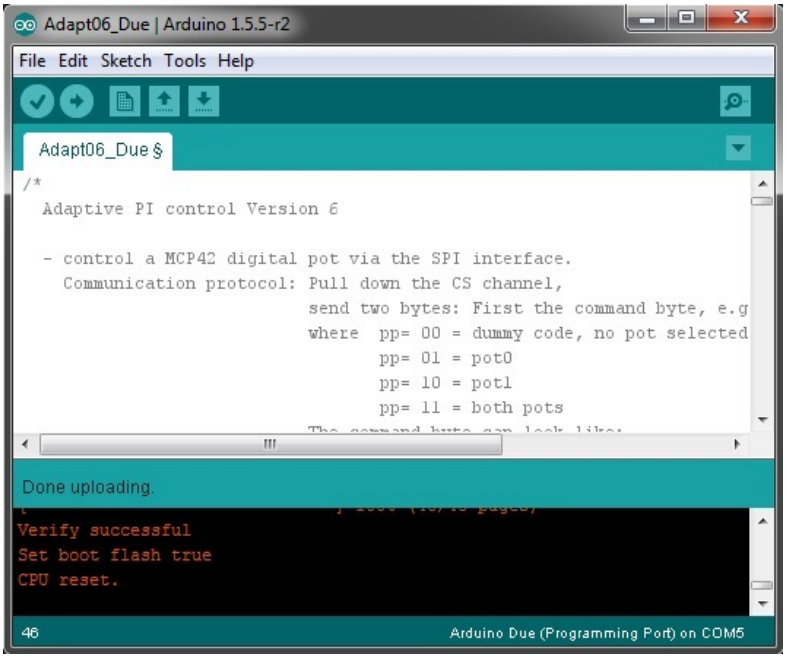

Figure 3.8.: Interface of the Arduino programming software that compiles the code and writes it to the Arduino. The software is freely available under `http://arduino.cc/en/Main/Software`

```
RegionParam = 2;                                                    2
digitalPotWrite(CommandByte, SecIntValue);                          3
digitalWrite(r1,LOW);                                               4
digitalWrite(r2,HIGH); }                                            5
```

The complete code of the `loop()` function as well as a separate function for changing the potentiometer resistance are given in App. B. Note that for constant signals all if-loops will be rejected, resulting in a minimum loop time and thus maximum responsiveness of the gain scheduling. The disadvantage of probing only shifts of the setpoint is that the gain is flipped only between adjacent gain regimes. The consequence is that for large and sudden jumps, for example from 0 to 100 %, the Arduino has to flip the gain from the first to the second to the third regime etc. and it thus takes more time to optimize the gain. This could be circumvented by including further if-loops to probe changes between all gain regimes which, however, adds a substantial amount of code and increases the loop time. We decided not to include these cases in the code since in the experiment the intensity is always ramped over at least a few milliseconds. We find that the gain adjustment between adjacent regime requires typically around 20 μs and

thus even five consecutive gain adjustments present no harm when designing intensity profiles for the experiment.

Tuning the gain parameters

Once the interaction between the Arduino and the circuit is set up we tune the gain parameters to optimum values. In practise, we ramp up both the integrator gain and the loop gain to values just below the onset of oscillations in steady state operation at very low intensities. The proportional gain is adjusted to an optimum value by observing the controller response to step functions at low intensities (as discussed in Sec. 3.3.4) such that the overshoot of the intensity signal is slightly dampened and the controller does not yet oscillate. However, the contribution of the proportional feedback to the PI feedback is always at least an order of magnitude weaker than the integral feedback.

3.3.3. Measuring the bandwidth

Having presented the control circuit and its interaction with the Arduino we will now examine the speed performance of the adaptive intensity stabilizer by measuring the bandwidth in all intensity regimes. We define the bandwidth as the frequency where the amplitude of a small setpoint modulation has decreased to half the amplitude of a modulation at very low frequencies. To do this, we add a small modulation on the setpoint by using the setpoint offset op-amp of the presented circuit as a summer (`IC2C`): A frequency generator is connected via a 10 kΩ resistor to one of the pins of a jumper (`JP1`). The output of the offset op-amp is thus the sum of the input from the frequency generator and the setpoint input. This allows us to apply a small modulation at any setpoint level while monitoring the resulting intensity modulation on an oscilloscope where we measure its amplitude. The general guideline for choosing the modulation amplitude is to have a sufficiently large intensity variation to observe it on a linear out-of-loop photodiode with a good signal to noise ratio while keeping the modulation small enough to ensure that we probe the bandwidth at a particular intensity level rather than across a range of intensities and thus different gains. Therefore, we decrease the amplitude of the modulation gradually from 60 mV for low intensities to 10 mV for highest intensities since setpoint variations with different offsets lead to different absolute intensity variations due to the logarithmic amplification of the monitoring photodiode.

An example measurement of the controller bandwidth at 1% of the maximum intensity

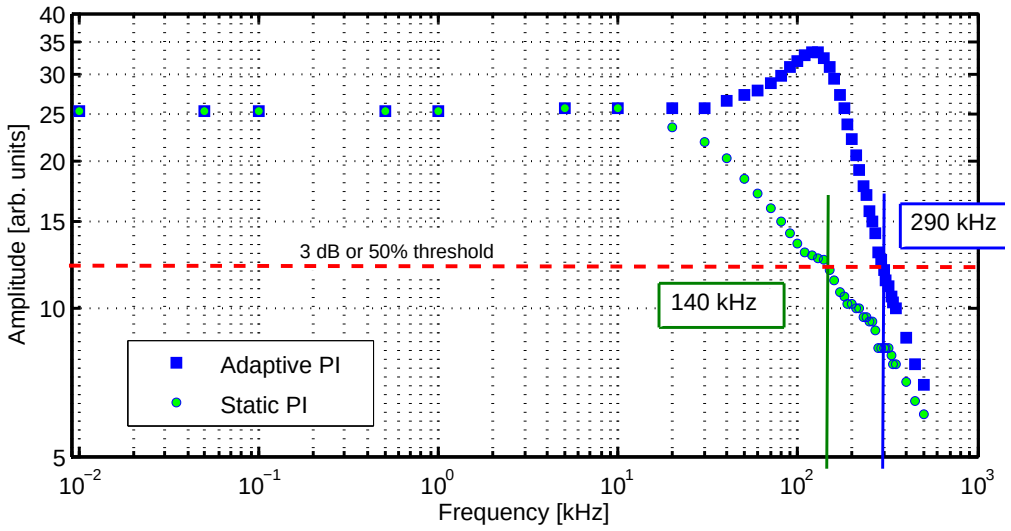

Figure 3.9.: Bandwidth measurement of the intensity stabilizer at 1% of the maximum intensity. For low frequencies the modulation amplitude is constant before it increases for the adaptive controller at higher frequencies, presumably due to a phase lag of the controller larger than π which changes the feedback sign. We observe the onset of ringing at ∼100 kHz. At 290 and 140 kHz the amplitudes of the adaptive and static PI controller respectively have decreased to half their initial levels, thus defining the bandwidth of the controller at this intensity level. The adaptive controller is operating in the second gain regime, giving it a substantially higher bandwidth compared to the static configuration.

is given in Fig. 3.9. We observe that the modulation amplitude for both the static and the adaptive controller remains constant for frequencies up to 10 kHz. Beyond this frequency the amplitude modulation of the static controller decreases steadily, crossing the 50% threshold that defines the bandwidth at 140 kHz. The adaptive controller, however, has higher integrator gain at this intensity level thanks to the gain schedule, and the modulation amplitude remains steady up 30 kHz. Beyond this frequency, the modulation amplitude increases by ∼30 % at 130 kHz. This so called servo bump is the result of a phase lag of the feedback loop larger than π for which the feedback sign is positive, resulting in a slight ringing at this frequency. This ringing implies an increased sensitivity of the controller at frequencies around 130 kHz at this intensity and in this gain regime. It need not be a problem except for the case of particularly high noise at this

frequency which we dismiss for our setup. The modulation amplitude decreases sharply beyond the servo bump, decreasing to 50 % at 290 kHz. Thus, the graph illustrates that the adaptive PI controller comprises a more than twofold larger modulation bandwidth than the compared static PI controller at an intensity level of 1 %. Note that it would be possible to increase the static integrator gain, too, in order to increase the bandwidth. However, this entails a gain which is too high at lower intensities where the controller would then ring strongly.

Measurements for different intensities

Based on the previous measurement routine we can evaluate the bandwidth across the entire range of intensities from 0.05 % of the maximum intensity level up to 60 %. The result is presented in Fig. 3.10 where we also compare the bandwidth of the Arduino based controller to a static PI controller that is optimized for low intensities[5]. For low intensities up to 0.1 % the Arduino based controller is essentially equivalent to the static PI controller and the bandwidth is above 300 kHz, far above the initially discussed requirements of at least 100 kHz bandwidth. As the intensity level increases, the gain of the photodiode decreases (as shown in Fig. 3.3) as does the gain of the static PI loop. Thus, for 0.1 % of the full power the bandwidth has already decreased to 200 kHz. As the intensity increases the bandwidth of the static PI feedback loop decreases further, while we note that the Arduino based PI controller is able to restore the bandwidth to more than 300 kHz as the integrator gain is adjusted. The black lines in the figure indicate flips of the integral gain according to the predefined gain schedule presented in the previous section. Note that the bandwidth increases every time that the integral gain is increased. Clearly, the gain schedule reduces and for low to intermediate intensities even cancels out the decreasing gain of the logarithmic photodiode. For laser intensities up to 10 % of the maximum power the bandwidth is above the required 100 kHz while for the static PI-controller this is already the case above 2 %. As the intensity level increases even further the bandwidth of the Arduino based controller decreases to 180 kHz at 10% and 45 kHz at 60 %. This decline is even more drastic for the static controller with 26 kHz at 10 % and 2 kHz at 60 %. The latter numbers illustrate that a static PI controller is barely suitable for an intensity stabilizer operating with a logarithmic photodiode.

[5] This is also the only feasible static PI controller that does not oscillate in any intensity regime. The static PI controller is activated by turning a switch on the front panel (`S1`) such that `R33` determines the integral gain rather than the digital potentiometer.

Intensities above 60 %

For intensities above 60 % it proved difficult to measure the regulation bandwidth with the described method. This was a result of a significantly decreased signal to noise ratio for the observation of a small modulation at high intensities. In this regime the intensity noise is larger due to the decreased photodiode sensitivity, which superposes a small intensity modulation. Therefore we cannot present an accurate bandwidth measurement in this regime. However, we can still infer that the controller operates at good speed from the step function measurements presented in Sec. 3.3.4 where a settling time after a 10 % setpoint change of $\sim$30 μs is observed at 90 % of the maximum intensity.

Declining bandwidth with increasing intensities

We attribute the overall bandwidth decline to three factors: Firstly, the bandwidth of the logarithmic photodiode itself declines for higher intensities as was shown in Fig. 3.2 from around 1 MHz for low intensities to 140 kHz for the highest intensities. Secondly, the gain of the AOM goes towards zero as the controller output signal sent to the AOM driver approaches the maximum value of 6 V. Lastly and probably least relevant, despite decreasing the modulation amplitude the absolute intensity modulation at high intensities is higher than for low intensities, resulting in a larger output swing from the controller which can be limited by the slew rate of the op-amps. However, it was not possible to decrease the amplitude modulation even further without substantially decreasing the signal to noise ratio of the observed signal. Still, the overall voltage swing of the controller output was small enough even at high intensities, below the slew rate values of the datasheets of the involved op-amps, making it an unlikely cause to limit the modulation bandwidth.

3.3.4. Response to step functions

A useful illustration to demonstrate the performance of the intensity stabilizer is to observe its response to a step function, where the controller has to follow a sudden change in the setpoint. While the previous bandwidth measurements are a very systematic way of assessing the regulation speed of the controller, step function measurements are very useful for optimizing the controller parameters as we can directly observe an overshoot of the regulated parameter as a consequence of a high gain. On the other hand, low gain leads to a smaller maximum gradient of the monitored intensity level. Additionally,

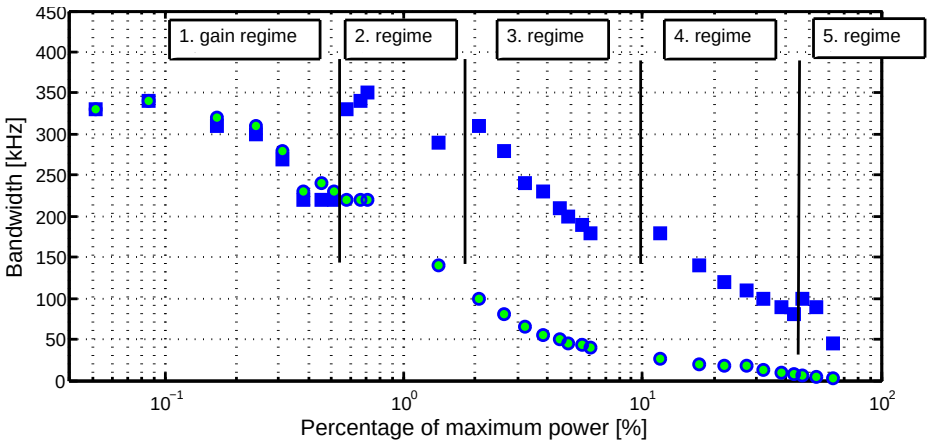

Figure 3.10.: Bandwidth of the adaptive PI-controller (blue) and the static PI-controller (green) over the entire intensity spectrum. While the bandwidth is similar for extremely low intensities the decreasing gain of the logarithmic photodiode causes the bandwidth of the static controller to decrease substantially while the adaptive controller counters this by adjusting the integrator gain as indicated by the black lines. The bandwidth decrease for very high intensities is mostly attributed to the decreasing bandwidth of the logarithmic photodiode in this regime, as shown in Fig. 3.4.

step function responses can illustrate the speed of a controller more intuitively than the previous bandwidth measurements. We present the response of the adaptive intensity stabilizer to step functions in all five gain regimes in Fig. 3.11, recorded with a linear, out-of-loop photo diode, and compare it to a static PI controller.

The graphs illustrate how the adaptive stabilizer maintains good responsiveness in all intensity regimes with particularly good performance at low intensities. This corresponds well to the previously presented bandwidth measurements. Unsurprisingly, the static PI controller shows good performance in this regime, too, but the settling time increases substantially for larger intensities while the adaptive controller settles within 30 μs even at highest intensities.

A small overshoot is visible for the adaptive PI controller at all intensity and is a consequence of the high controller gain. Decreasing the gain would eliminate the overshoot but also decreases the maximum slope of the intensity change and thus decrease the controller bandwidth in this regime which is necessary to cope with high frequency intensity noise. Thus we consider a small overshoot of less than 10 % acceptable. We

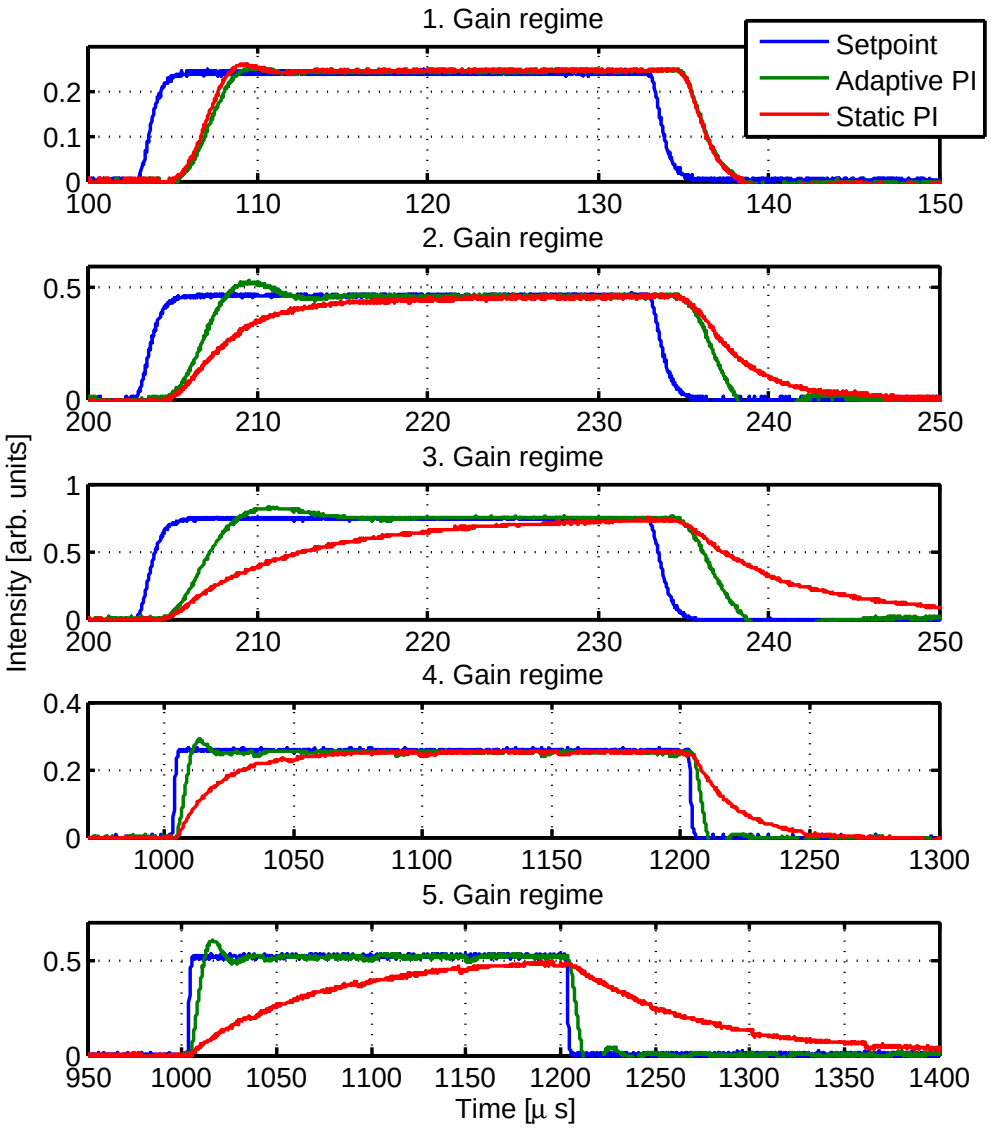

Figure 3.11.: Response of the adaptive intensity controller (red) to a sudden small setpoint change (blue) in all five gain regimes. The response of a static PI controller is shown for comparison (green). The setpoint is increased by 10 % except for in a), and changes as follows: **a)** from 0 to 0.1 %, **b)** from 1.23%, **c)** from 4.5 %, **d)** from 16.5 %, **e)** from 81%. While the adaptive and static configuration show similar performance at low intensities the settling time of the static PI controller increases drastically for higher intensities where the adaptive PI controller benefits strongly from the gain scheduling.

did not notice any effect of the adaptive gain scheme on the steady state noise.

Summary of the speed evaluations

In summary the Arduino based intensity stabilizer has an excellent bandwidth of at least 100 kHz up to 10 % of the maximum intensity. This includes the intensities required to generate the lattice depths in which strongly correlated quantum systems are studied and even the lattice depths used for Raman-sideband cooling. From step functions we observe that even at highest intensity values the settling time is around 30 μs. Thus we can conclude that the adaptive PI controller provides good responsiveness over the entire intensity spectrum. This is not the case for a static PI controller, as both the bandwidth measurement and the step function response curves reveal.

3.3.5. Measuring the accuracy

Just as important as controlling the laser intensity with high bandwidth is to do so with high resolution, that is very close to the desired value, and high precision, that is with little fluctuation around that value. We measure the precision as follows: On an out-of-loop photodiode (Thorlabs, PDA36A) with a bandwidth of more than 1 MHz we monitor the intensity such that a 10 V output signal corresponds to the highest intensity level, i.e. 100 %. This means that 1 mV of signal corresponds to 0.01 % of the maximum intensity. We then set the intensity to four different magnitudes and evaluate the noise of the linear photodiode by monitoring the AC component of the photo diode signal on an oscilloscope. We note that the inherent noise of the photodiode ('dark noise'), i.e. the noise when no light is incident, is around 1 mV. The results of the noise measurments are presented in Table 3.1. For very low intensities the noise around 1 mV, corresponding to a precision better than 0.01 %. At the same time, we can set the intensity level in steps of at least 0.1 mV change of the photodiode signal, corresponding to a resolution of 0.001%. To give an intuition for these numbers consider that the highest intensity of the optical lattice laser beams is 15 W. Then our precision at 0.1 % of the maximum intensity, that is at 15 mW, amounts to 1.5 mW and we the resolution at which the intensity can be set is at least 0.15 mW. These values for the resolution and precision are upper limits and might be well below these values as the low intensity resolution of the logarithmic photodiode is higher than of the linear photodiode. Indeed, the noise measured at lowest intensities has the same amplitude as the photodiode inherent noise

Intensity level	Noise on out-of-loop linear photodiode [mV]	Corresponding intensity noise
0.1 %	1	0.01 %
1 %	4	0.04 %
10 %	20	0.2 %
100 %	40	0.4 %

Table 3.1.: Precision of the intensity stabilizer in different intensity regimes. At the lowest intensities the noise seems to be below the monitoring photodiode inherent noise ('dark noise') of 1 mV. Here, intensity noise of 0.01 % is thus only an upper limit. At higher intensities the effect of the declining gain of the logarithmic photodiode become visible and the precision of the controller decreases.

that is present even in complete darkness. Only at higher intensities the controller noise exceeds the darkness noise. At the highest intensity the precision has decreased to 0.4 %. This is acceptable since the corresponding lattice depths do not need to be controlled with very high precision. We note that in all regulation regimes the noise on the logarithmic photodiode is constant. This means that the circuit minimizes the error signal with constant precision across the entire range of intensities and only the varying sensitivity of the logarithmic photodiode is responsible for decreased precision at high intensities.

We conclude that the Arduino based intensity stabilizer has good precision and offers good intensity resolution in the relevant intensity regimes. Combined with the previously investigated bandwidth it thus satisfies the requirements described at the beginning of the chapter. Having examined the core purpose of the controller we now turn to possible other applications to demonstrate the flexibility of the developed circuit board.

3.4. Output directly from the Arduino

We described in Sec. 3.3.1 how the controller is designed such that the Arduino can read in the photodiode signal, the setpoint and the PI output signal of the controller. In addition we can also switch the output of the intensity stabilizer between the analog PI output and the digital-to-analog output (DAC) of the Arduino via a solid-state relay. This enables further applications of the stabilizer, three of which we will discuss here.

Completely digital control

Firstly, it is possible to generate the controller output digitally via the Arduino. In this case, both the setpoint and the photodiode signal are read by the Arduino and it computes an output, for example based on a PID algorithm. In general, the bandwidth of this digital PID control is limited by the DAC and ADC conversion time as well as the processing time of the Arduino. Thus, this control mode is not suitable for the high bandwidth intensity stabilization that is the main objective of the presented controller but is sufficient for a slower process such as temperature stabilization. In this case, a temperature measurement circuit is required (for example a Wheatstone bridge in conjunction with an INA126 precision op-amp) that provides a signal via the photodiode input of the circuit. A setpoint could either be provided via the setpoint input of the controller or it is stored on the Arduino. The accuracy of this digital PID controller is then limited by the resolution of the temperature measurement relative to the 10-bit resolution of the Arduino ADC as well as the maximum DAC output resolution of 12-bit.

Sample and hold mechanism

Secondly, we can use the Arduino to read in and store the output of the analog PI controller. This can be triggered externally via one of the digital inputs. The controller output is flipped to Arduino output via the relay and a triggering pulse on another digital input signals the Arduino to output the stored value for a predefined amount of time[6]. This scheme enables us to pulse the intensity for short periods of a few microseconds that are shorter than the settling time of the analog PI controller. Pulsing the intensity means that we can 'flash on' the optical lattice for short times in order to observe Kapitza-Dirac scattering, a powerful method to calibrate optical lattices [32].

Prewarming the AOM

Lastly, we can use the Arduino output to warm up the AOM before operation. This can be necessary as in some cases the deflection properties of the AOM change as the AOM warms up during operation. Such a warming sequence could be as follows: While the mechanical shutters are still closed, the intensity stabilizer is triggered to output generic 6 V, thus warming up the photonic crystal of the AOM. To do this, again we switch from the analog output to the digital output and another triggering pulse signals

[6] Depending on the loop time of the Arduino code this is accurate by a few microseconds.

the Arduino to give out a predefined constant voltage. After a few seconds the photonic crystal of the AOM has warmed up and is thus ready for operation. Now the setpoint is set to zero and the controller output is switched back to analog PI-output, which will be zero now. Next, the mechanical shutter is opened and the setpoint is set to the desired value. If the time between turning off the controller output and ramping up the analog output after opening the shutters is short enough, that is well below a second, the photonic crystal is still warm. Therefore, this sequence can counter the temperature dependent deflection behavior of an AOM.

These examples outline the high flexibility of the developed Arduino based intensity stabilizer. They all benefit from the fact that the Arduino as part of the controller facilitates implementing logic operations that are usually more complicated in completely analog circuits.

This chapter

We have presented an Arduino based intensity controller that consists of a logarithmic photodiode, a home built circuit board and an Arduino microprocessor that is mounted on this board. The Arduino schedules the gain of the feedback loop to encounter the varying gain due to the photodiode. The result is a modulation bandwidth above 100 kHz for intensities below 10 % while the controller settling time is around 30 μs even at the highest intensities. Moreover, the control accuracy and precision at low intensities benefits from the high sensitivity of the photodiode in this regime and amounts to less than 0.1 and 0.01 % respectively, allowing good control over the depth of the optical lattice in our experiment.

4. Conclusion

We have presented an adaptive PI-controller consisting of an analog feedback loop and an Arduino micro-controller that schedules the controller gain via a digital potentiometer. We decided to develop the controller as an analog-digital hybrid only after initial tests with completely digital control based on an FPGA board revealed that common ADC and DAC converters with conversion times of 2 μs and thus loop times of at least around 5 μs would be too slow as part of a feedback chain that should opperate with at least 100 kHz bandwidth. The first steps of this project included the setup of an FPGA based PID-controller and experiments with different FPGA boards. Eventually, an analog circuit combined with a digital potentiometer appeared to us as a feasible and straightforward approach to remedy problems with the gain properties of our feedback loop. Here, Arduino micro-controllers were a valuable tool since they are easy to program, cheap and reasonably functional with sufficient processing speed and freely available libraries and code examples. We believe that combining analog circuitry with such micro-controllers has great potential for applications in the lab as they allow for easy implementation of logic operations such as a gain schedule that are otherwise very complicated to realize in analog circuits. Thus, our controller harnesses the advantages of both the analog and digital realm: high bandwidth combined with logic operations. Given the broad range of Arduino micro-controllers and additional Arduino circuitry ('shields') such as motor drivers, Ethernet and WiFi connectors, many more applications in the lab seem possible, including temperature stabilization of laser diodes and interlock systems.

We presented and described the first adaptive PI-controller in our lab, but for the experiment at least three controllers will be required to regulate all lattice beams. Beyond that the controller may also be used to stabilize the intensity of other beams though it might not be necessary to implement a logarithmic photodiode in those cases due to a smaller dynamic range. In this case, the controller can conveniently be used as a static PI controller. We hope that this thesis will be a useful guide to construct and set up other copies of the intensity stabilizers in the near future, to provide the Quantum Gas

Microscope with excellent intensity control.

Bibliography

[1] K. B. Davis, M.-O. Mewes, M. R. Andrews, N. J. van Druten, D.S. Durfee, D.M. Kurn, and W. Ketterle. Bose-Einstein Condensation in a Gas of Sodium Atoms. *Phys. Rev. Lett.*, 75:3969, 1995.

[2] M. H. Anderson, J. R. Ensher, M. R. Matthews, C. E. Wieman, and E. A. Cornell. Observation of Bose-Einstein Condensation in a Dilute Atomic Vapour. *Science*, 269:198, 1995.

[3] F. Dalfovo, S. Giorgini, L. P. Pitaevskii, and S. Stringari. Theory of Bose-Einstein condensation in trapped gases. *Rev. Mod. Phys.*, 71:436, 1999.

[4] W. Ketterle and M. W. Zwierlein. Making, probing and understanding ultracold Fermi gases. *Proceedings of the International School of Physics 'Enrico Fermi', Course CLXIV, Varenna, 20 - 30 June 2006*, page 1, 2006.

[5] I. Bloch. Ultracold quantum gases in optical lattices. *Nature Physics*, 1:23, 2005.

[6] M. Greiner and S. Fölling. Optical lattices. *Nature*, 453:736, 2008.

[7] S. Inouye, M. R. Andrews, J. Stenger, H.-J. Miesner, D. M. Stamper-Kurn, and W. Ketterle. Observation of Feshbach resonances in a Bose-Einstein condensate. *Nature*, 392:151, 1998.

[8] R. Feynman. Simulating Physics with Computers. *IJTP*, 21:567, 1982.

[9] I. Bloch, J. Dalibard, and S. Nascimbene. Quantum simulations with ultracold quantum gases. *Nature Physics*, 8:267, 2012.

[10] D. Jaksch, C. Bruder, J. I. Cirac, C. W. Gardiner, and P. Zoller. Cold Bosonic Atoms in Optical Lattices. *Phys. Rev. Lett*, 15:3108, 1998.

[11] M. Greiner, O. Mandel, T. Esslinger, T. W. Haensch, and I. Bloch. Quantum phase transition from a superfluid to a Mott insulator in a gas of ultracold atoms. *Nature*, 415:39, 2002.

[12] T. Esslinger. Fermi-Hubbard Physics with Atoms in an Optical Lattice. *Annu. Rev. Condens. Matter Phys.*, 1:129-152, 2010.

[13] R. Jördens, N. Strohmaier, K. Günter, H. Moritz, and T. Esslinger. A Mott insulator of fermionic atoms in an optical lattice. *Nature*, 455:204, 2008.

[14] M. Köhl, H. Moritz, T. Stöferle, K. Günter, and T. Esslinger. Fermionic Atoms in a Three Dimensional Optical Lattice: Observing Fermi Surfaces, Dynamics, and Interactions. *Phys. Rev. Lett.*, 94:0031–9007, 2005.

[15] Y. Miroshnychenko, D. Schrader, S. Kuhr, W. Alt, I. Dotsenko, M. Khudaverdyan, A. Rauschenbeutel, and D. Meschede. Continued imaging of the transport of a single neutral atom. *Optics Express*, 11:3498, 2003.

[16] K. D. Nelson, X. Li, and D. S. Weiss. Imaging single atoms in a three-dimensional array. *Nature Physics*, 3:556, 2007.

[17] W. S. Bakr, J. I. Gillen, A. Peng, S. Fölling, and M. Greiner. A quantum gas microscope for detecting single atoms in a Hubbard-regime optical lattice. *Nature*, 462:74–77, 2009.

[18] J. F. Sherson, C. Weitenberg, M. Endres, M. Cheneau, I. Bloch, and S. Kuhr. Single-atom-resolved fluorescence imaging of an atomic Mott insulator. *Nature*, 467:68–72, 2010.

[19] C. Weitenberg, M. Endres, J. F. Sherson, M. Cheneau, P. Schauß, T. Fukuhara, I. Bloch, and S. Kuhr. Single-spin addressing in an atomic Mott insulator. *Nature*, 471:319-324, 2011.

[20] T. Fukuhara, A. Kantian, M. Endres, M. Cheneau, P. Schauß, S. Hild, D. Bellem, U. Schollwöck, T. Giamarchi, C. Gross, I. Bloch, and S. Kuhr. Quantum dynamics of a mobile spin impurity. *Nature Physics*, 9:235, 2013.

[21] J.-S. Bernier, C. Kollath, A. Georges, L. De Leo, F. Gerbier, C. Salomon, and M. Köhl. Cooling fermionic atoms in optical lattices by shaping the confinement. *Phys. Rev. A*, 79:061601, 2009.

[22] M. Lubasch, V. Murg, U. Schneider, J. Ignacio Cirac, and M.-C. Banuls. Adiabatic Preparation of a Heisenberg Antiferromagnet Using an Optical Superlattice. *Phys. Rev. Lett.*, 107:165301, 2011.

[23] D. Greif, T. Uehlinger, G. Jotzu, L. Tarruell, and T. Esslinger. Short-Range Quantum Magnetism of Ultracold Fermions in an Optical Lattice. *Science*, 340:1307, 2013.

[24] A. J. Kerman, V. Vuletić, C. Chin, and Steven Chu. Beyond Optical Molasses: 3D Raman Sideband Cooling of Atomic Cesium to High Phase-Space Density. *Phys. Rev. Lett.*, 84:439, 2000.

[25] Y. S. Patil, L. M. Aycock, S. Chakram, and M. Vengalattore. Nondestructive imaging of an ultracold lattice gas. *Phys. Rev. A*, 90:033422–1, 2014.

[26] B. J. Lester, A. M. Kaufman, and C. A. Regal. Raman cooling imaging: Detecting single atoms near their ground state of motion. *Phys. Rev. A*, 90:011804–1, 2014.

[27] K. Åström and T. Hägglund. *PID Controllers: Theory, Design and Tuning.* Instrument Society of America, 2010.

[28] J.G. Ziegler and N. B. Nichols. Optimum settings for automatic controllers. *Transactions of the ASME*, 64:759, 1942.

[29] Isomet. Application note AN1106, Maximizing AO Diffraction efficiency. `http://www.isomet.com/App-Manual_pdf/Maximizing%20DE.pdf`.

[30] B. Fröhlich, T. Lahaye, B. Kaltenhäuser, H. Kübler, S. Müller, T. Koch, M. Fattori, and T. Pfau. A two-frequency acousto-optic modulator driver to improve the beam pointing stability during intensity ramps. *Rev. Sci. Instrum.*, 78:043101, 2007.

[31] Analog Devices. AD8304 Logarithmic converter (datasheet). `http://www.analog.com/static/imported-files/data_sheets/AD8304.pdf`.

[32] P. L. Kapitza and P. A. M. Dirac. The reflection of electrons from standing light waves. *Mathematical Proceedings of the Cambridge Philosophical Society*, 29:297, 1933.

A. Electronic circuits, drawings and pictures

A.1. Logarithmic photodiode

The logarithmic photodiode used by the intensity stabilizer is comprised of the circuit presented in Fig. A.1. The key principle is the logarithmic amplification of the photodiode current via `AD8304`. As discussed in Sec. 3.2 it is possible to adjust both the slope and the zero intercept of the logarithmic amplification via periphery resistors.

A.2. Power supply board

The adaptive-PI controller board is powered by a separate power supply board housed in the same module. A schematic of this board is presented in Fig. A.2. Located on the board are three 600 mA fuses to prevent the controller from drawing excessive currents in case of a shortcircuit. The board provides ±15, 9, 5 and 3 V which are generated via a transformer, a bridge rectifier and several voltage regulators.

A.3. The test setup

An optical test setup was constructed for this work on which the presented measurements were performed. A photo of this test setup is shown in Fig.A.3. An 852 nm diode laser provides light up to 150 mW power that is directed through an optical isolator to avoid harmful back reflections into the laser diode. Subsequently, the light passes through a $\lambda/2$-waveplate, the AOM and a polarizing beam splitter (PBS) to stabilize the polarization of the beam. Part of the light is then directed to the logarithmic photodiode for regulation and to a linear photodiode for out-of-loop monitoring of the intensity.

A.4. The front panel

The presented Arduino based intensity stabilizer is hosted in a Schroff module in order to fit into the electronics racks in the lab. We have designed a front panel where the necessary signals are connected via BNC ports and some of the potentiometers of the circuit can be adjusted. A sketch of the front panel is presented in Fig. A.4.

The front panel allows for switching between an adaptive, Arduino-based PI configuration and a static PI configuration. In both cases, the P-part of the feedback loop can be adjusted via a potentiometer on the front panel. If adaptive control is enabled, the I-part of the feedback loop is adjusted by the Arduino and the gain regime is indicated via the LED array, where red indicates the highest gain. In the case of static control the I-part can be adjusted via a front panel potentiometer.

The photodiode input signal can be scaled to match the range from 0 to 10 V via an offset and an amplification control on the front panel.

A USB port on the front panel can be used to program the Arduino.

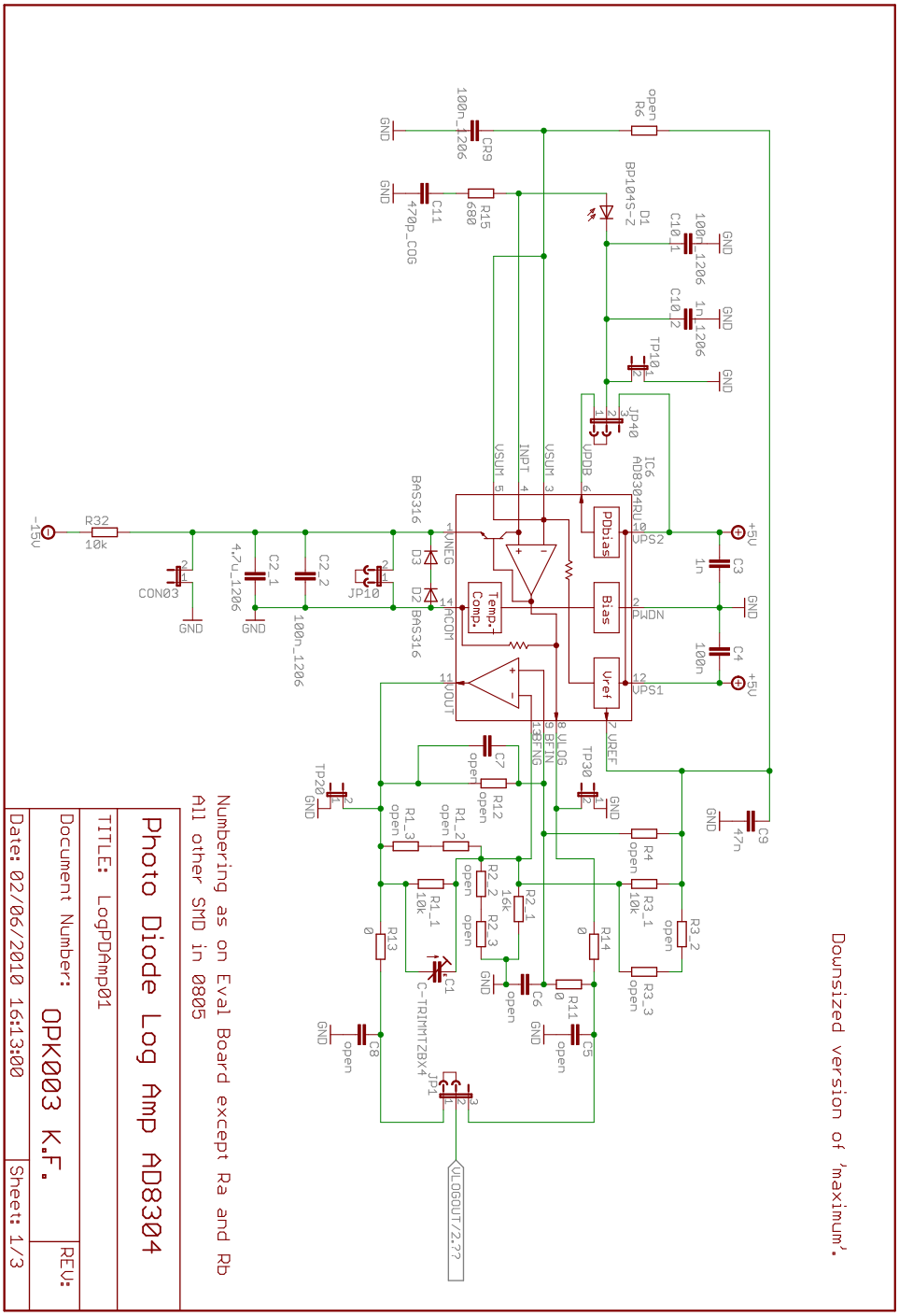

Figure A.1.: The circuit board around the photodiode and the logarithmic amplifier developed at the Max-Planck-Institute in Garching.

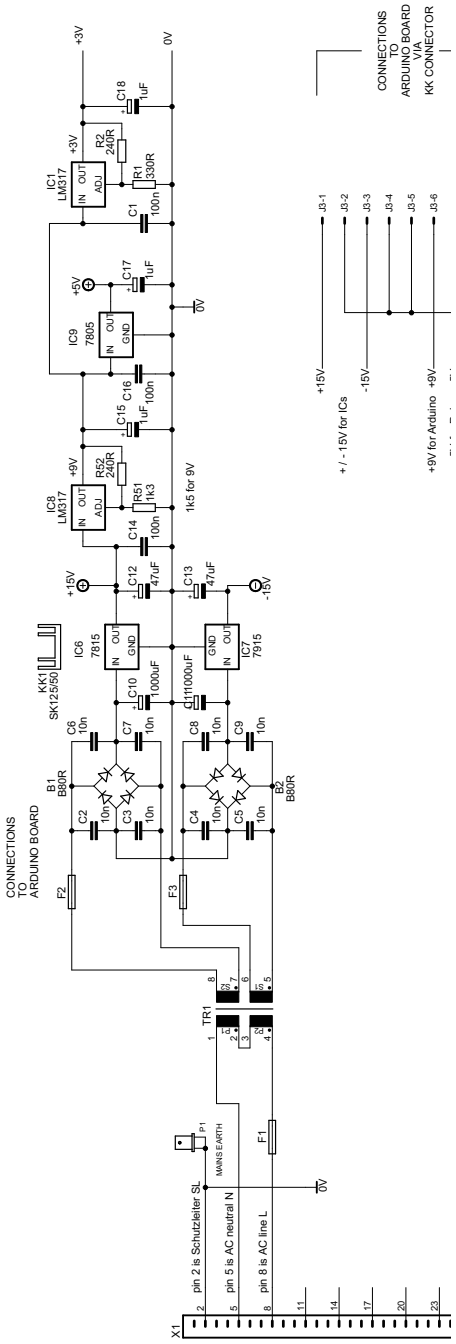

Figure A.2.: The power supply board for the adaptive PI-controller, delivering ±15, 9, 5 and 3 V.

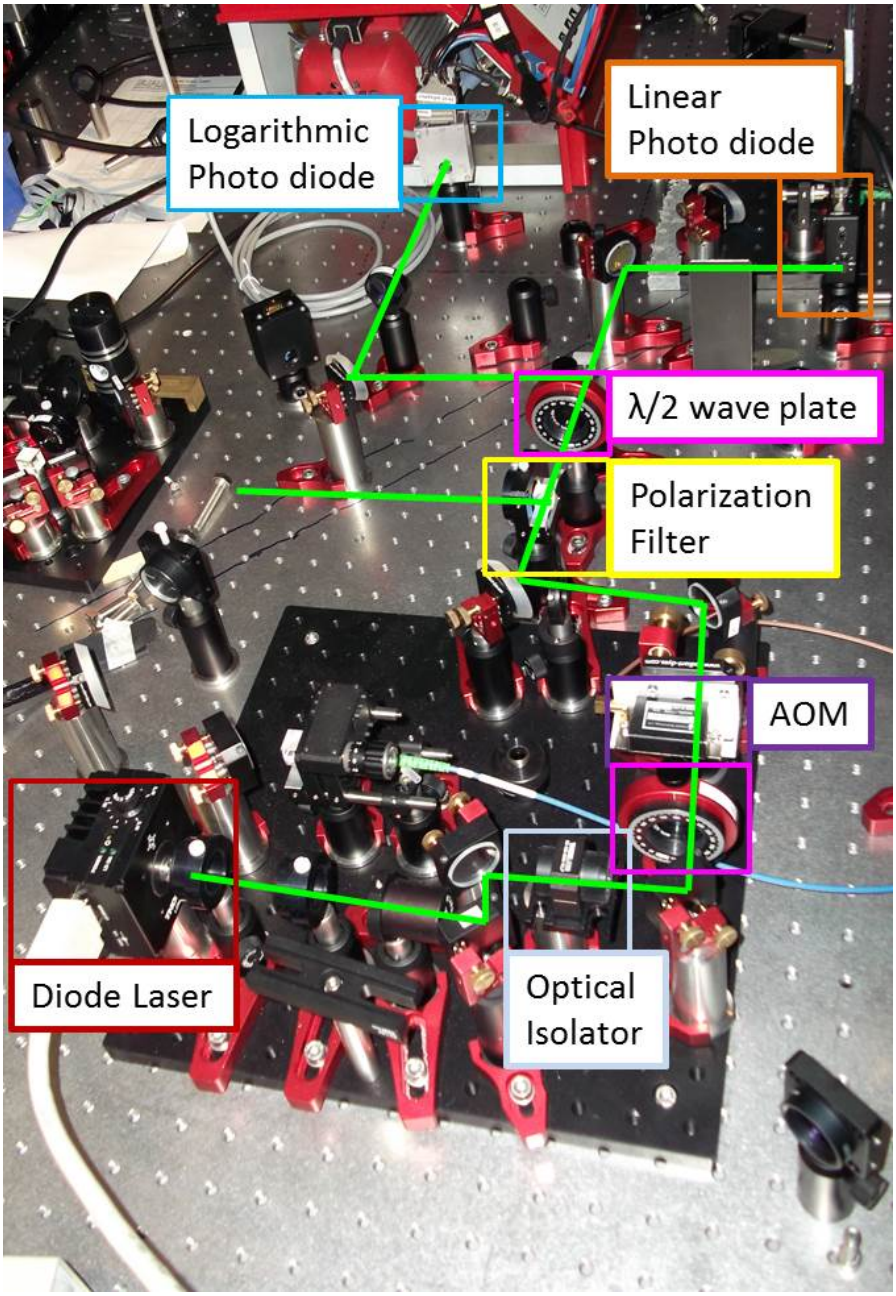

Figure A.3.: The optical setup constructed for testing the intensity stabilizer.

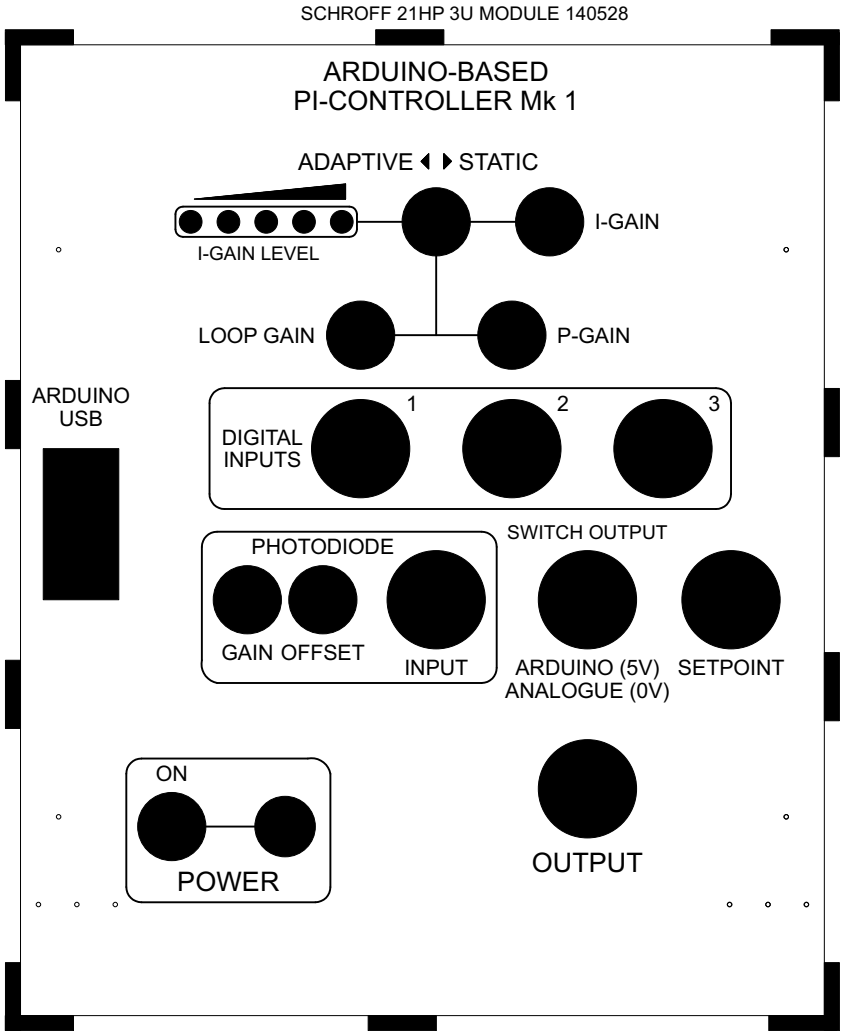

Figure A.4.: The front panel of the Schroff module housing the adaptive intensity stabilizer. The largest black circles indicate BNC ports while the smaller ones are potentiometers for tuning, switches and LEDs.

B. The Arduino code

Here we present the complete code executed by the Arduino Due to enable gain scheduling. The program was named *Adapt06* as it is the sixth version since the first gain scheduling code was written.

On line 1 we include the SPI library that provides the function `SPI.transfer(<byte>)` which we require to write a new resistance value to the digital potentiometer. We will not go into the specific functioning of this library but from the discussion of the SPI communication between the Arduino and the digital potentiometer we know already that the key concept is to send 8-bit packets on the `SPI MOSI` pin while providing a clock on the `SPI CLK` pin of the Arduino Due. On line 2 we define the chip select pin where a pull-down selects the potentiometer for resistance adjustment and line 3 defines the Command Byte, the first byte that the SPI protocol sends to the digital potentiometer to indicate which resistor channel configuration is chosen[1]. Next we define the gain schedule itself, on lines 5 to 20, where the previously mentioned dead zones are also included. The values of the gain schedule were obtained from careful calibration. The front panel of the intensity stabilizer includes 5 LEDs to indicate the gain regime. These LEDs are driven by the Arduino via its digital output pins, defined on lines 25 to 29. Subsequently, the `setup()` function is programmed, where the required Arduino pins are configured[2]. The function `led_startup()` plays a small startup sequence on the LEDs and can be omitted here. Lastly, the SPI functions are initialized and the SPI communication speed is set to 9600 bits per second.

```
#include <SPI.h>                                                    1
const int slaveSelectPin = 13;                                      2
int CommandByte = 19;                                               3
// Region Values                                                    4
```

[1] The binary notation of '19' is '00010001' where the last two bits indicate the resistance channel. '00' selects no channel, '01' selects channel 0, '10' selects channel 1 and '11' selects both channels. The stabilization circuit uses only channel 0.

[2] Note that only digital pins have to be configured while the analog in- and output pins can be operated immediately.

```
int FirstIntValue = 180;
int SecIntValue = 65;
int ThirdIntValue = 45;
int FourthIntValue = 18;
int FifthIntValue = 5;
// Region definitions
int DeadZone = 40;
int HighPowerDeadZone = 10;
int FirstRegionLower = 350;
int FirstRegionUpper = FirstRegionLower + DeadZone;
int SecRegionLower = 530;
int SecRegionUpper = SecRegionLower + DeadZone;
int ThirdRegionLower = 730;
int ThirdRegionUpper = ThirdRegionLower + DeadZone;
int FourthRegionLower = 985;
int FourthRegionUpper = FourthRegionLower + HighPowerDeadZone;

int RegionParam =1;
int measured;
// region leds
int r1 = 2;
int r2 = 3;
int r3 = 4;
int r4 = 5;
int r5 = 6;
void setup() {
  pinMode (slaveSelectPin, OUTPUT);
  pinMode (r1, OUTPUT);
  pinMode (r2, OUTPUT);
  pinMode (r3, OUTPUT);
  pinMode (r4, OUTPUT);
  pinMode (r5, OUTPUT);
  digitalWrite(r2,LOW);
  digitalWrite(r3,LOW);
  digitalWrite(r4,LOW);
  digitalWrite(r5,LOW);
  led_startup();
        SPI.begin();
  Serial.begin(9600);
  digitalWrite(r1,HIGH); }}
```

This code is carried out once when the Arduino is powered up or reprogrammed. Sub-

sequently, the microprocessor carries out the `loop()` function until power is shut down or it is reprogrammed. The code below shows the array of if-loops that check if the setpoint has shifted from one gain regime to an adjacent gain regime. If any loop is accepted the gain is adjusted via the function `digitalPotWrite` (defined below the `loop()` function) that updates the resistance of the digital potentiometer. Additionally, the region parameter is updated and the corresponding LEDs on the front panel are turned off and on.

```
void loop() {
  measured = analogRead(0);

  if (measured > FirstRegionUpper && RegionParam == 1) {
    RegionParam = 2;
    digitalPotWrite(CommandByte, SecIntValue);
    digitalWrite(r1,LOW);
    digitalWrite(r2,HIGH);}

  if (measured < FirstRegionLower && RegionParam == 2) {
    RegionParam = 1;
    digitalPotWrite(CommandByte, FirstIntValue);
    digitalWrite(r2,LOW);
    digitalWrite(r1,HIGH);}

  if (measured > SecRegionUpper && RegionParam == 2) {
    RegionParam = 3;
    digitalPotWrite(CommandByte, ThirdIntValue);
    digitalWrite(r2,LOW);
    digitalWrite(r3,HIGH);}

  if (measured < SecRegionLower && RegionParam == 3) {
    RegionParam = 2;
    digitalPotWrite(CommandByte, SecIntValue);
    digitalWrite(r3,LOW);
    digitalWrite(r2,HIGH);}

  if (measured < ThirdRegionLower && RegionParam == 4) {
    RegionParam = 3;
    digitalPotWrite(CommandByte, ThirdIntValue);
    digitalWrite(r4,LOW);
    digitalWrite(r3,HIGH);}
```

```
  if (measured > ThirdRegionUpper && RegionParam == 3) {
    RegionParam = 4;
    digitalPotWrite(CommandByte, FourthIntValue);
    digitalWrite(r3,LOW);
    digitalWrite(r4,HIGH);}

  if (measured < FourthRegionLower && RegionParam == 5) {
    RegionParam = 4;
    digitalPotWrite(CommandByte, FourthIntValue);
    digitalWrite(r5,LOW);
    digitalWrite(r4,HIGH);}

 if (measured > FourthRegionUpper && RegionParam == 4) {
    RegionParam = 5;
    digitalPotWrite(CommandByte, FifthIntValue);
    digitalWrite(r4,LOW);
    digitalWrite(r5,HIGH);}
}

int digitalPotWrite(int CommandByte, int value) {
  digitalWrite(slaveSelectPin,LOW);
  SPI.transfer(CommandByte);
  SPI.transfer(value);
  digitalWrite(slaveSelectPin,HIGH);
}
```

C. The LabVIEW FPGA interface

To test the intensity stabilizer with various setpoint profiles we have developed a small LabVIEW FPGA program, executed by a National Instruments FPGA board. The board features analog and digital voltage outputs that we used for driving setpoint profiles, providing mV voltage resolution between -10 and 10 V and a temporal resolution of 2 μs. A screen shot of this software is shown in Fig. C.1. The interface allows the user to define voltage sequences with up to 15 different consecutive voltage values, for each of which the duration can be defined with microsecond accuracy. A trigger signal for an oscilloscope is provided at the beginning of a sequence on a digital output. The software can either output a predefined sequence or a constant voltage. The corresponding LabVIEW files are stored on the QI-group drive under `Q:\Electronics+Datasheets\HomeBuiltDevices\028_Adaptive Intensity Stabilizer (Jakob)\LabVIEW FPGA`.

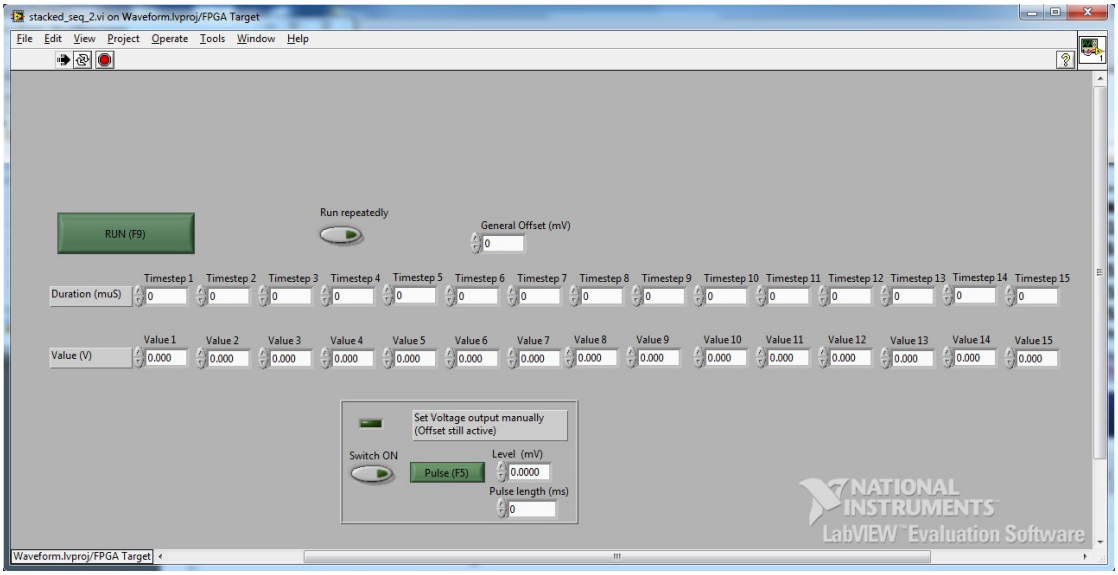

Figure C.1.: The LabVIEW FPGA interface for driving setpoint profiles. The program can either output a defined sequence or a constant voltage between -10 and 10 V.

D. A guide for setting up the intensity stabilizer

This section provides a step-by-step guide to set up the adaptive PI-controller and optimize its operation. To be able to tune all parameters of the controller it is necessary to remove it from the rack and open it in order to access the potentiometers on the board. The module can be powered with the same connector that powers AOM drivers (12 pins, three of which provide the mains) and thus can be operated separately during the setup. Additionally, a tunable voltage source (ideally the LabVIEW FPGA interface, since it can output well defined sequences and a trigger signal) and an oscilloscope are necessary. With this equipment at hand, the following steps should be carried out.

(i) *Scaling the controller input signals:* First, one has to check if the photodiode and the setpoint signal cover the same range. Assuming a setpoint range from 0 to 10 V, the photodiode signal has to be scaled to that range. Amplification of the photodiode signal is controlled via `R30` ('photodiode gain' on the front panel) and can be monitored via the corresponding test point `TP11`. In some cases the minimum photodiode signal deviates from 0 V, e.g. due to stray light, which can be encountered by offsetting the signal by connecting `JP4` and adjusting `R9` ('photodiode offset' on the front panel). It is generally good to ensure that the scaled photodiode signal covers the setpoint range down to 0 V because otherwise the controller does not regulate at very small setpoint values and is likely to oscillate if it operates at the threshold between this regime and the regulated regime.

The same scaling options are available for the setpoint input but if that is provided by the experiment control system it should not be necessary to amplify anything. In this case, one should make sure that the signal at `TP8` is the same as the input signal by adjusting `R27` to unity gain with `JP1` unplugged.

(ii) *Scaling the Arduino input signals:* Since the Arduino Due cannot handle input

signals above its operating voltage of 3.3 V one has to ensure that the photodiode and setpoint signal sent to the Arduino are scaled to that range. Again, both signals can be amplified via `R29` (photodiode) and `R28` (setpoint) and an offset can be added via `JP3` and `R8` (photodiode) and `JP2` and `R2` (setpoint). Since the Arduino schedules the gain via the setpoint signal it is important to not change the setpoint scaling anymore after optimizing the gain schedule.

Additionally, the analog PI output is also sent as an input to the Arduino and again this range from 0 to 6 V has to be mapped to the voltage range of the Arduino via `R36` (amplification) and `R43` and `JP6` (offset). The circuit board also allows for output from the Arduino in which case the range of the DAC converters (0.04 to 2.7 V) has to be scaled to the range from 0 to 6 V.

(iii) *Tuning the static controller parameters:* Both in the adaptive and the static PI-control configuration, it is necessary to manually optimize the static controller parameters, that is the proportional gain (`R32`) and the loop gain (`R44`). To do this, first one has to ensure that the loop gain is sufficient to generate more than 6 V controller output. Set the controller to the static PI-control mode and the setpoint to the maximum level. At this point, the controller output should be below but close to 6 V. If it is smaller than that, increase the loop gain. If it is larger than 6 V, i.e. it ran up to the maximum output, it means that the controller cannot steer the photodiode signal high enough to diminish the error signal. In this case the photodiode amplification is probably insufficient and should be adjusted such that the controller output at the maximum setpoint value is on the order of 5.5 V. Having tuned the loop gain in this manner, next the setpoint should be set to the low intensity regime (where the feedback loop has highest gain and is thus most prone to oscillations) and the proportional (`R32`) and static integral gain (`R33`) can be optimized. Here, a good strategy is to set both to the minimum and then increase the integral gain below the threshold of oscillations. Then do the same for the proportional gain. In practice we noted that there is only a small regime with a notable but not yet oscillating proportional contribution to the controller output and that the error signal is minimized mostly via the integral component.

The controller should now work reasonably well as a static PI-controller throughout the entire intensity regime. Next, we turn to the Arduino to enable and optimize gain scheduling.

(iv) *Adjustments to the Arduino code:* The Arduino code presented in this thesis can be found on the QI-group drive under `Q:\Electronics+Datasheets\HomeBuiltDevices \028_Adaptive Intensity Stabilizer (Jakob)\Arduino`. The gain schedule defined there should be already close to the optimum since it was calibrated carefully with the test setup. However, the gain parameters of the whole feedback loop change with different photodiode illumination and AOM efficiency and it might be necessary to calibrate the gain schedule again across the entire intensity spectrum. In order to ensure that the gain schedule is executed but does not lead to oscillations one should scan the setpoint slowly from 0 to 10 V and ensure that the LEDs on the front panel indicate the change of the gain regime. If that is not the case, check that the Arduino is properly connected to the board and that the signals sent to it are in the correct range. At the same time, monitor on the oscilloscope that the controller does not oscillate in any of the regimes.

Optimizing the gain schedule can be a bit lengthy because one has to adjust a particular gain parameter, upload the new gain schedule to the Arduino and then ensure that the controller does not oscillate in any part of the corresponding gain regime. Note that due to the inbuilt hysteresis the gain regimes are flipped at different setpoints depending on whether the setpoint is increased or decreased. Thus, one also has to ensure that no oscillations occur when scanning the setpoint in both directions.

After carrying out these steps the controller should operate properly with scheduled gain. In the case of difficulties with the setup, it can be very helpful to probe the error signal on `TP21` which is zero if the controller operates properly. Here, one can also observe well if the controller starts to oscillate. One can ensure that the digital potentiometer works properly by probing the corresponding pins (5 and 7 of `U1`) with an Ohmmeter where one can observe changes of its resistance.