UPPSALA
UNIVERSITET

Electron gun vacuum system control

Project in Embedded Systems

Simon Gollbo

2017-03-17

Supervisors: Uwe Zimmermann, Ping Wu

Abstract

The purpose of this project is to design a control system for a vacuum system in the Ångström laboratory clean room. This entails interfacing a microcontroller to industrial grade electronics and a touch screen interface for user input. A universal PID-controller is also designed, to which the user can input feedback gains using the touch screen. The project was prototyped on a breadboard after which a PCB was designed, ordered and hand-mounted. The resulting PCB, after some minor adjustments, worked to specifications.

Contents

1. Introduction

1.1. Background

In the clean room of the Ångström Laboratory there is an electron gun system which is used to deposit thin films of metal onto a substrate. The electron gun uses an electric field to accelerate electrons which form an electron beam of high power. The electron beam is then directed using magnetic fields into a crucible that holds the metal to be deposited. The high energy of the electron beam vaporizes the metal and the metal vapor shoots upwards toward the substrate target.

To avoid oxidation and contamination of the samples, the process needs to be carried out in a vacuum chamber. The vacuum chamber for the system at the Ångström Laboratory utilizes two pumps, an air pump and a cryo pump. The air pump is able to bring the chamber from atmospheric pressure down to $10^{-2}\ mbar$. At this pressure, the air pump is unable to lower the pressure further and the cryo pump is used. The cryo pump works by moving a portion of gas from the chamber to a separate chamber. The separate chamber is then cooled down using liquid helium, which compresses the gas inside. The compressed gas is then moved to a storage tank. The cryo pump is able to bring the pressure in the vacuum chamber down to $10^{-8}\ mbar$. The air pump is required due to the limited volume of the cryo pump storage tank. The cryo pump storage tank fills quickly if the cryo pump is used at atmospheric pressure.

1.2. Purpose of the project

The purpose for this project is to upgrade the control system for the vacuum system. Currently, there is no mechanism for ensuring that the user performs the correct sequence of actions to utilize the vacuum system. An example of erroneous usage is incorrectly turning on the cryo pump at atmospheric pressure which leads to the cryo pump storage tank filling up. If this were to happen, the cooling of the system has to be turned off and the gas has to be allowed to heat up before being ventilated. The system then has to cool back down. This entire process takes several days to complete during which time the electron gun cannot be used.

Also, the current control system is spread out over several controls. This project also aims to gather all controls in a single control device.

The project also aims to implement a universal PID-controller. The scope of the project does not include implementing the PID-controller in the vacuum system, but control of the metal crucible could be controlled using this type of controller. As such, the universal PID-controller enables further work on the control system in the future.

1.3. Project specifications

The control system is to use a touch screen interface with a graphical user interface which displays the current state of the system. The various valves and pumps are to be controlled via the touch screen. The touch screen is run at 3.3V.

The main logic of the control system is implemented on a microcontroller. The microcontroller is interfaced with the touch screen and reads where the user is touching the screen as well as communicating what the screen is to display.

The valves and pumps have existing interfaces for control, however they operate at 24V. This means that an interface is required between the logic levels of the microcontroller, which is run at 3.3V, and the external 24V controls. To protect the microcontroller and touch screen, some form of insulation with regards to the external interfaces is required.

The system should disallow or warn the user when a selected action is not appropriate, for example if the user should attempt to open an erroneous combination of valves.

The universal PID-controller should also be controlled via the touch screen interface. The feedback gains of the PID-controller should be customizable, as well as the reference signal level.

1.4. Project planning

1.4.1. Hardware

The first objective to be achieved is to get the microcontroller online and programmable on a breadboard for prototyping and testing purposes. Related to this is also establishing correct procedure for communication between the microcontroller and the touch screen and getting the screen to display basic objects. Also, the touch interface should be made to output values when pressed.

Next, the touch interface coordinates should be calibrated such that the coordinates read for a press on the touch interface corresponds to the same position on the graphics display.

To insulate the microcontroller from the digital output side of the system mechanical relays are to be used. These should be tested. On the digital input side of the system, optocouplers should be used. Similarly, these should be tested.

When the components of the system have been prototyped and tested, a PCB is to be designed and ordered along with components. The components are to be soldered by hand to the PCB. When the PCB has been mounted, it is to be tested.

1.4.2. Software

The first software objective is to program a hello world for the microcontroller to ensure functionality.

Functions for interfacing with the touch screen should be written, essentially an application programming interface (API). A set of functions that provide functionality for communicating with the screen, a set of functions that allow objects and text to be written on the screen and a set of functions for interfacing with the touch interface, for example to retrieve touch coordinates, form the main components of the API.

Using these functions, graphical user interfaces (GUIs) should be designed for the vacuum control system and the universal PID-controller respectively. The vacuum control system GUI

should display the current state of the system and visualize a system overview. It should also have menu screen for the various components of the system. The universal PID-controller GUI should have a main screen displaying the current feedback gains as well as the current reference level. It should also contain numerical input menu screens for changing the feedback gains and the reference level.

A main routine is also to be written. This main routine should coordinate the subroutines that handle the touch interface, the drawing to the screen as well as the input and output logic. It should poll the touch interface for user input and draw the various GUI screens on the screen depending on what the user inputs.

2. Working principles

2.1. Touch screen

The touch screen to be used in the project is a so-called 4-wire resistive touch screen. This type of screen consists of two sheets layered on top of each other, with the top layer being flexible. An example of this configuration is shown in Fig. 2.1.1, where the bottom sheet consists of glass, to form a stable base, and the top sheet consists of a flexible plastic film.

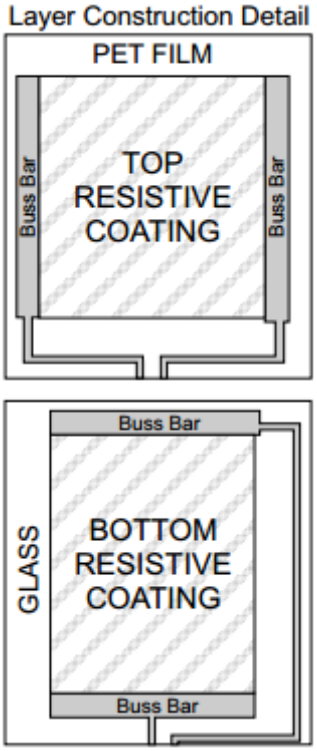

Figure 2.1.1: Example of a 4-wire touch screen configuration. (Sparkfun)

The two sheets are coated in a resistive material. One layer is outfitted with connection terminals along either the north and south sides of the layer or the east and west sides of the layer. The second layer is outfitted with terminals in whichever configuration was not applied to the first layer. (Sparkfun)

To operate the screen, a voltage is applied across the north terminal to the south terminal or across the east terminal to the west terminal. When the sheets are brought together they start conducting essentially forming a voltage divider with one resistor on the top sheet and a second resistor on the bottom sheet. By reading the voltage on one of the unused terminals, the ratio of the read voltage value and the full applied voltage value yields the ratio of where the screen was pressed, in a single dimension. The procedure can then be repeated by applying the voltage across whichever north and south or east and west configuration was not used. The resulting pair of measurements corresponds to where the screen was pressed. (Sparkfun)

2.2. Microcontroller

A microcontroller is a system-on-chip consisting of a microprocessor with additional peripherals such as general purpose input/output (GPIO) ports, timer modules, memory modules, analog to digital converter (ADC) modules et cetera.

The microcontroller is programmed by uploading compiled code to its memory. As the microcontroller boots up, it reads the program memory and starts executing instructions.

2.3. PID-controller

A Proportional Integrating Differentiating-controller, or PID-controller, is a feedback controller which takes as input a reference level. The difference between the reference signal and the system output is called the control error. A correctly designed controller attempts to generate control signals such that the control error goes to zero. Worded differently, the controller compares the reference level to the current output level of the system and generates a control signal, which is sent to the system with the aim to control the system to the reference level.

As the name suggests, a PID-controller consists of three parts, a proportional feedback, an integrating feedback and a differentiating feedback. These three parts each have a feedback gain assigned to them, essentially controlling how much each of the three parts affects the control signal. The process of determining sufficiently good values for the feedback gains is called tuning.

The proportional feedback is, as the name suggests, proportional to the control error. The proportional feedback gain is simply a constant with which the control error is multiplied. It is very direct in its acting, essentially constantly telling the system to move in the direction of the reference signal.

The integrating feedback keeps track of the control error over time. When properly designed, it ensures that the system, if stable, is controlled to control error zero in the steady state. Compared to the proportional feedback, the integrating feedback is slower but is able to compensate for small errors, which the proportional feedback may not be able to, given enough time.

The differentiating feedback is related to fast changes in the control error. If the system suddenly jerks, the differentiating feedback is able to provide the control signal with an extra boost to compensate for the quick change in system output.

3. Implementation

3.1. Overview of the system

The system consists of a microcontroller which can be interfaced to industrial grade electronics. The system state can be controlled by a user via a touch screen interface which also displays a graphical user interface. The system is also able to read analog voltage signals as well as being able to output pulse width modulated signals that can be filtered to achieve outputting of analog voltage signals.

3.2. Hardware and components

3.2.1. Microcontroller

For prototyping purposes the ATMega328 in a PDIP capsule was used on a breadboard. For the PCB the ATMega644 in a QFP capsule was chosen for its additional GPIO ports to allow for additional flexibility in the finished project.

3.2.2. PCB

The PCB was designed using a single layer for components and two copper layers. Power nets and ground nets are supplied via polygons and stitched with vias to ensure stable voltage levels. The polygons do not flow under the external input and output components to further isolate the microcontroller from externalities. The traces connecting the output channels were made thicker to be able to withstand higher currents.

3.2.3. Relays

To output digital signals to external systems running at higher voltages and/or currents than the microcontroller can withstand, mechanical relays are used. A mechanical relay typically has three connector leads, one common, one normally closed and one normally open. Additionally a mechanical relay has coil leads. The coil acts as an electromagnet, mechanically switching a contact between the normally closed lead and the normally open lead. The microcontroller output port drives the base of a bipolar transistor which can switch the relay coil current and toggle the relay. This enables the microcontroller to send an output to an external device without having to interface with it directly.

3.2.4. Optocouplers

To input digital signals from external systems running at higher voltages and/or current than the microcontroller can withstand, optocouplers are used. Optocouplers are optically isolated and contain an LED on the input side and a phototransistor on the output side. When an input switches the LED on, the phototransistor begins to conduct. This enables the microcontroller to detect that an input has been switched on without it having to interface with the external system directly.

3.2.5. Touch screen

The touch screen used is a 4-wire resistive touch panel with a 320 by 240 pixel LCD display. It has a variety of interfaces for communicating with the screen driver. SPI was chosen for this purpose.

3.2.6. Power supply

During prototyping, power was provided via the USB-port of the programming computer. USB outputs a maximum of 5V which was not sufficient to drive the mechanical relay coils. To test these, an external variable power supply was used, which could provide up to 20 V.

Since the mechanical relays need a higher voltage than the microcontroller, the PCB was outfitted with a switched voltage regulator. As input, the board is supplied with 24V, which is regulated down to 3.3V for the microcontroller and touch screen.

3.2.7. USBasp programmer

To transfer the compiled program from the computer to the microcontroller memory, a USBasp programmer clone was used.

3.3. Integrated Development Environment (IDE)

The software for the microcontroller was written in C. To compile the code, AtmelStudio was used. AtmelStudio provides microcontroller specific project creation where necessary definitions are automatically included.

The text editor in AtmelStudio is decent, but I opted to use Sublime Text, detailed below, for writing the software. AtmelStudio was quick to recognize whenever the file being edited in Sublime Text was saved and was immediately ready to compile.

3.4. Development tools

3.4.1. Sublime text

As mentioned above, Sublime Text was used to write the code. Sublime Text has many handy features, such as multi-selection editing and auto completion, which made it my primary choice. The program is responsive and feels lightweight which was also found advantageous.

3.4.2. Git

Git was used to backup and version code. It was also handy when working from different locations, being able to synchronize files.

3.4.3. AVRDUDESS

The software used to communicate with the USBasp programmer was AVRDUDESS, which is a graphical user interface for the programming API AVRDude. It has a built in library of presets for various different programmers and is able to detect what type of microcontroller it is connected to.

3.4.4. KiCad

To design the PCB, the open source program KiCad was used. KiCad comes equipped with tools to draw circuit schematics, footprints and PCB schematics as well as some additional tools.

3.5. Implementation

3.5.1. Hardware

The project hardware had two main phases, a prototype phase and a PCB phase. The prototype consisted of an ATMega328 in a PDIP package mounted on a breadboard. The ATMega328 was connected to the programmer and the touch screen using external cables. An image of the setup is shown in Fig. 3.5.1. The figure shows the programmer circuit in the red and white rectangle and the microcontroller under the wires in the blue and white rectangle. To the right in the figure, the touch screen is shown. In the top right, the power supply that was used for testing the mechanical relays is shown.

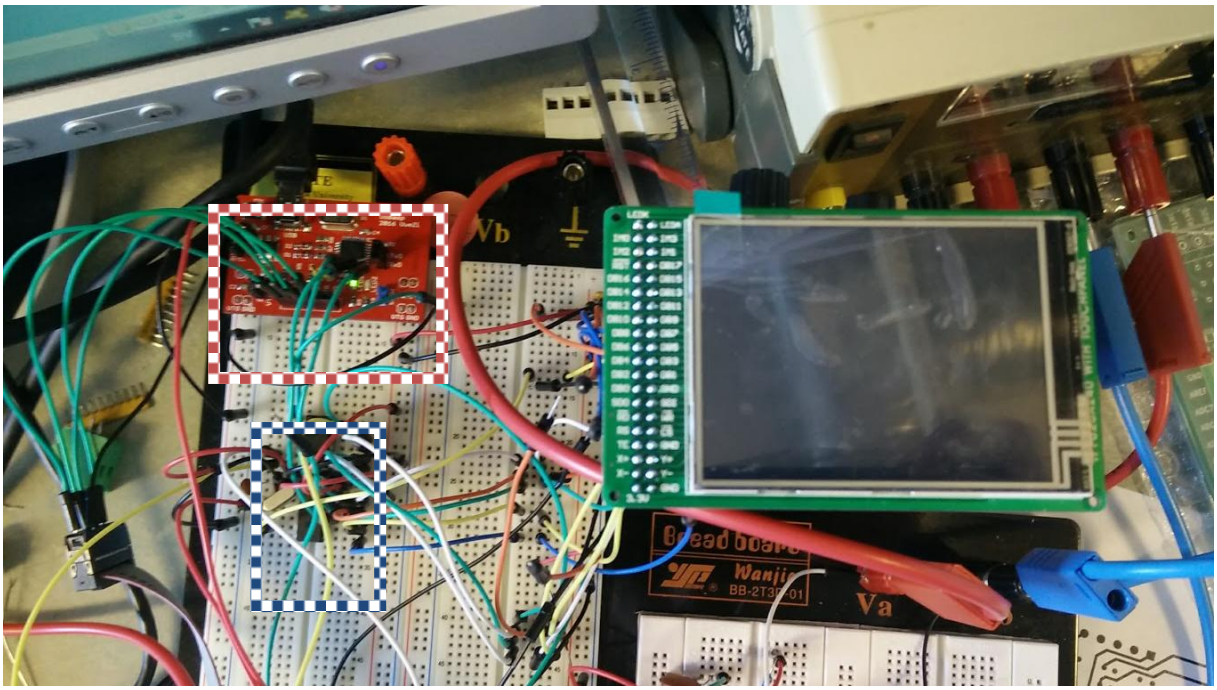

Figure 3.5.1: Prototype breadboard setup. The programmer is shown inside the red and white rectangle in the top left. The ATMega328 microcontroller is under the wires in the blue and white rectangle below the programmer.

When the PCB design was finished and the PCBs arrived, work was started to mount components on the board. In Fig. 3.5.2 the PCBs without mounted components is shown.

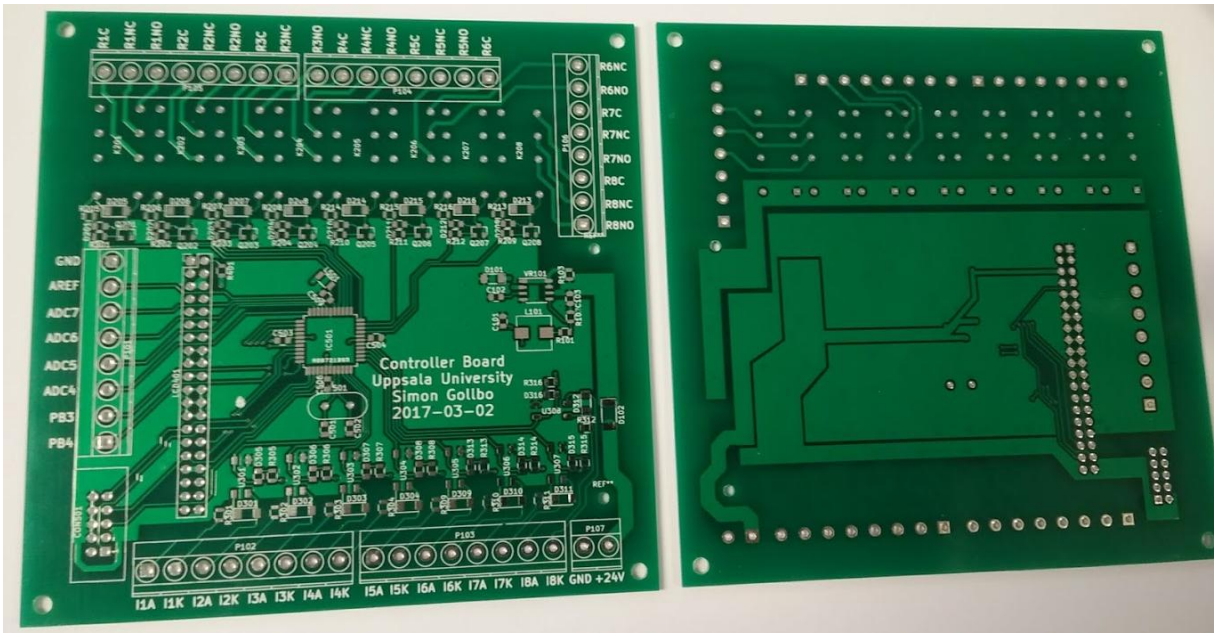

Figure 3.5.2: Project PCBs without components mounted.

As mounting progressed, it was discovered that an inductor had been misplaced in the design of the power supply unit. Due to time constraints it was decided against ordering new PCBs for the moment and instead rewiring the affected pins by cutting away connections that were incorrect and soldering Teflon insulated wires to make the missing correct connections. Also, it was opted to use a larger capacitor between the ground plane and the 24V feed line than was in the original design. To mount this, the plastic layer was scraped off and the capacitor was soldered directly to the exposed copper. Fig. 3.5.3 shows the resulting power supply unit on the PCB.

Figure 3.5.3: The power supply circuit after alteration to correct for the erroneous placement of the inductor.

No further errors were discovered during mounting. The finished PCB is shown without the screen mounted in Fig. 3.5.4 and with the screen mounted in Fig. 3.5.5.

Figure 3.5.4: The PCB without the touch screen mounted

Figure 3.5.5: The PCB with the touch screen mounted

3.5.2. Software

The software was implemented in C.

To communicate with the touch screen, command codes are sent along with data. The touch screen was poorly documented, but in its datasheet there was a snippet of code which outlined the initialization.

These types of touch screens are commonly used together with Arduino systems and sample code for interacting with the screen could be found written in C++. The C++ code was used as inspiration when writing the same functionality in C, with some adjustments where appropriate. (adafruit)

The basic communication was established with the screen and the ability to modify a specific pixel was verified. Using the pixel modification as a primitive, an API was written with functions to draw simple objects on the screen, for example rectangles and text.

Additionally, a small library was written containing functions related to PID-control. The library contains functions for reading analog values which can be used for the feedback as well as the actual control loop that does the calculations.

4. Results and discussion

The aim of the project was to design a control system that is able to interface with industrial grade electronics. Using mechanical relays and optocouplers, the interface to industrial grade electronics is achieved without exposing the microcontroller to high voltage levels.

The system uses a touch screen that a user can use to interface with the system. This also enables the user to customize the PID-controller gains, in accordance with the project specification.

Throughout the project, various problems have been encountered. Debugging using an oscilloscope to find out determine whether some unexpected behavior is a result of hardware or software has been very useful. Hardware problems have usually taken the form of something being incorrectly connected. Software problems have usually taken the form of either typos not caught by the compiler or flawed program logic.

Perhaps the most difficult to debug problem was related to the fuse settings of the ATMega644. By default, some of the pins on the ATMega644 are reserved for a JTAG interface that enables programming of the device, which overrides their regular GPIO functionality. Fortunately, Uwe had already encountered this behavior and the problem was easily solved by changing the fuse settings.

A problem encountered with the PCB design was an incorrect connection of an inductor in the power supply. The error was discovered when mounting components on the PCB and due to time constraints it was not an option to redesign the PCB and order new ones. The problem was solved by cutting into the PCB, severing the incorrect connections, and adding insulated wires to connect the correct pins.

5. Conclusions

The control system performs according to the specifications set and the project can be considered successful.

My previous experience with microcontrollers was highly useful along previous general experience with coding and specifically coding in C. Even though I have never written C++, my previous programming experience allowed me to understand C++ code written for the Arduino which made coding the touch screen API easier.

For the PCB design, my previous experience with Altium Design was highly useful. Transitioning from Altium Design to KiCad required only minor adjustments and while KiCad is not an as fully fledged electronic CAD environment as Altium Design, it was fully sufficient for this project.

6. References

(n.d.). Retrieved March 16, 2017, from Sparkfun: https://www.sparkfun.com/datasheets/LCD/HOW%20DOES%20IT%20WORK.pdf

adafruit. (n.d.). Retrieved March 17, 2017, from Github: https://github.com/adafruit/Touch-Screen-Library/blob/master/TouchScreen.cpp

Glad, T., & Ljung, L. *Reglerteknik, Grundläggande teori.*

7. Appendix

Perhaps some code can be put here