NSF Lab Furnace Control System

PROJECT PLAN

Team 47
Client & Advisor: Dr. Gary Tuttle

Team Members
Nick Brylski - Systems Engineer
Jeremy Hartl - Hardware Engineer / Report Manager
Adam Matthews - Embedded Systems Engineer
Kevin Lang - Hardware Engineer
Chris Pohlen - Software Engineer / Gitlab monitor

https://sdmay19-47.sd.ece.iastate.edu/team.html

Revised: 12/4/18 Revision 3.0

Table of Contents

List of Figures

List of Tables

List of Definitions

OTC - Omega CN616 PID Temperature Controller. A device that can set and monitor up to 6 independent temperature channels.

MFC - Mass flow controller. A device that sets and monitors the rate of gas flow by measuring the mass of the gas.Oxidation - The forming of an oxide layer, commonly silicon dioxide, on top of a substrate.

Doping - The injection of chemicals into a material to create more electrons or holes

DAC - Digital to analog converter

ADC - Analog to digital converter

RS-232 - asynchronous serial communication standard used to transmit data over distance

I2C - synchronous serial communication standard , 2 wire

SPI - synchronous serial communication standard ,3 wire

1 Introduction

1.1 Acknowledgement

Senior design team 47 would like to thank our client and advisor Dr. Gary Tuttle. He has provided us with great insight and project materials. We would also like to thank Dr. Meng Lu, the current instructor for EE 432, and PhD student Harsh Gaonkar, a lab TA for EE 432.

We would also like to thank Texas Instruments and Analog Devices for providing us free samples for this project.

We'd also like to thank our TA Tim Lindquist for providing valuable input on our project.

1.2 Problem Statement

The NSF lab contains four furnaces that are used for the oxidation and doping of silicon wafers in EE 432: Microelectronic Fabrication. Three of these furnaces are operational. During operation, the state of any furnace is determined by two categories of parameters, temperature and gas flow rate. Each furnace has 3 temperature zones and either 1 or 3 gas channels. Temperature is controlled by two Omega CN616 PID Temperature Controllers (OTC) that can control set points and time profiles for up to 6 independent zones each, which can be seen in appendix 4.3.1. One controls 6 zones across two furnaces, and the other controls the 3 zones of the third. If the fourth furnace becomes operational, it will presumably be controlled by the second OTC. Gas flow is controlled by mass flow controllers (MFC) with one MFC per channel. Currently, set points and time profiles for the OTCs are controlled by three onboard buttons, and all feedback is displayed on 7 seven segment display digits. Each MFC has a dedicated control knob and seven segment display, as seen in appendix 4.3.2.

The control segment of this system can be greatly improved. Simply setting the temperature on a single zone requires a lengthy sequence of semi-ambiguous button presses that is prone to user error. Setting time profiles is even more complex. The controls for gas flow are simple, but there is no way to automate changes in flow rate.

The solution that we will implement will automate both the temperature control and gas flow control with a microcontroller. This microcontroller will be able to set and read temperature from the temperature controller. It will also be able to create profiles; e.g ramps 4 celsius/ min up to 1000c then hold for 15 hours--then ramp back down at 3 celsius/min to room temp. These profiles will include setting and shutting off the gas flow at certain time points. All temperature and gas flow information will be read on a mounted display connected to the microcontroller. A program on a separate computer, also connected to the microcontroller, will allow the user to navigate through and control each of the 4 furnaces.

1.3 Operating Environment

The operating environment for our product is the NSF lab in the Applied Science Complex. Our device, while interfacing with a temperature controller, will not come into contact with the 1000°C+ furnace. The user interface however, will be used on a weekly basis by students, TA's and researchers for years to come. This means that we will need to choose components that can last for many years. The display that we will show this information on will also need to be in robust housing to prevent anyone from bumping into it or getting scratched.

1.4 Intended Users and Intended Uses

The intended users for our NSF Furnace Control System are students, researchers, professors, and teaching assistants.

A heavy portion of the furnace usage is for the academic purposes of Iowa State University's EE 432 course. This Microelectronic Fabrication Techniques course consists of approximately 5 labs with 5 students and 1 teaching assistant per lab. These labs meet every week and last between 3-4 hours. Most students using the equipment have no previous experience with the NSF furnace. The heavy usage by inexperienced students requires our design to be user friendly and intuitive.

Researchers and professors who have previous experience using the furnace will be using it for longer, more precise processes. This kind of usage will desire more control over the equipment along with a better automated control. That way it can run on weekends without requiring the professor/researcher to come in and make adjustments.

1.5 Assumptions and Limitations

Assumptions:

- We will not need to do any PID tuning of the OTC's, but our project will allow for tuning commands to be sent.
- We will not need modify any existing thermocouples on the OTC's
- Has to withstand continuous usage and run 24/7
- The funding for the project is fairly large, we should have no problem getting all the hardware we need

Limitations:

- Can't be readily tested due to heavy lab usage and the immobility of the furnace
- Must be compatible with existing power supplies and hardware rack
- Will need to be compatible with 5V logic of MFCs
- will need to be compatible with RS232 protocol of OTCs
- The furnace will not be usable for testing until after Thanksgiving and Spring breaks, when the class finishes using the furnace.

- This means that the final phase of testing and installation may not happen until around April 2019

1.6 Expected End Product and Other Deliverables

The end product that we will be delivering to our client will be a user friendly interface to control the OTC's and MFC's. This will be accomplished with a GUI's, a computer running the GUI, an Arduino Mega 2560, and an Arduino shield and Python API of our own design. One GUI will be an always-on status readout running on a computer and displayed on a screen that can be mounted to the furnace equipment, and the other will run on a laptop and will be used to control the furnace settings. Both will use a serial interface to communicate with the Arduino that will intelligently pass commands to and read results from the OTC's and MFC's. In order to successfully communicate with these devices, the shield for the Arduino will include a digital-to-analog converter (DAC) and an RS232 transceiver, as well as the necessary cable connectors.

The GUI will include temperature settings and readouts. It will also have gas flow values for each of the furnaces. The control interface will allow users to set temperatures, timers, and temperature profiles.

The accuracy of the furnace control system is limited by the Omega Temperature controller and Mass Flow Controller (MFC). The Omega Controller has an accuracy rating of ± .2% or ± 2°C. The Mass Flow Controller is rated at an accuracy of one percent. The chosen DAC that will communicate with the MFC is rated at .5% which will allow the MFC to retain its accuracy at about one percent.

Additionally, several useful documents will be delivered to our client. For the user interface, a brief, descriptive document will be delivered. This document will contain basic operation instructions. However, the user interface as a whole will be very intuitive and most users will not need documentation to figure out the controls. Additionally, the Design Document and Project Plan will be available to the client. These documents are for the client to review and understand the process that we went through when designing our control system. To allow our client to find express concerns with our process and overall design, these documents are constantly available to him long before we deliver our final project.

The final completed, end product will be delivered by week 13 of spring semester (April 8-12).

2 Proposed Approach and Statement of Work

2.1 Objective of the Task

The objective of our project is to redesign the unintuitive and outdated interface between the NSF lab's furnaces and the people using them.

We will be doing a considerable amount of software development in Python and C to accomplish this task. The Python will be used to create a GUI and C will be used to program the Arduino microcontroller.

Besides software, much hardware evaluation and testing will need to be done. We will be working with multiple ICs (a RS232 transceiver and DAC) that will need to be integrated with the arduino and OTC/MFC. This testing will use common hardware like oscilloscopes and multimeters to make sure they are operating correctly.

2.2 Functional Requirements

The product must be able to:

- control and display gas flow
- control and display furnace zone temperature
- automate temperature and gas flow changes according to a given profile
- Be able to run 24/7
- include a data logging feature to allow a temperature time series to be written out to the user

2.3 Constraints & Considerations

One of the major constraints with this project is that the seven mass-flow controllers (MFCs) only operate off 0-5V, this means we need to be able to apply a 0-5V to them and read 0-5V off them. The computer is not suitable for this because it does not have any integrated ADCs and can only accept 3.3V digital inputs. Using an ADC would be complicated because this ADC would have to convert a variable 0-5V signal into a digital signal at 3.3V. For the DAC, it would have to convert a 3.3V signal into 0-5V variable, another odd case.

The better solution then is to use an Arduino Mega. The mega has an integrated 16-channel 10-bit ADC that would work perfect for the 7 MFCs. It only has a 2-channel DAC so we would not be able to use this, but we could still easily find an 8-channel DAC that could convert our 5V logic into variable 0-5V. Using an external ADC would also be quite a bit more complicated than using the analogRead(pinNum) on the arduino.

Another constraint on the project is that we must control two Omega temperature controllers. This means we need two RS232 converter chips and two serial ports. The computer only has one serial port so if we controlled the OTCs directly from the Pi we would need to use additional IC's or use USB to RS232 converters.

One standard lab protocol that we will need to adhere to is to have a display that is constantly being updated with new data and is always on and showing that updated data. This is important for lab users to be able to easily and quickly review the states of the furnace and MFCs. Lab users will sometimes be wearing gloves or holding delicate materials, so not having to press buttons, or go through a procedure to access real-time status is important.

For the safety of the lab users, it is also very important to have the display of our control system show warnings and errors when appropriate. If a furnace is hotter than expected or gases are flowing at an unusual rate, not only is the device under fabrication at danger, but so are the lab users. User safety is extremely important to us and our final product is set to meet the safety standards of organizations such as IEEE.

2.4 Previous Work And Literature

Our project project consists of integrating hardware and software to control the Omega temperature controller (OTC) and mass flow controllers. Our most important piece of literature in interfacing with the OTC is its manual. This manual describes the wiring for the RS-232 interface, the operating procedures, and a list of RS-232 commands. These commands can be used to access all of the OTC's functionality.

Omega has already developed windows software that showcases everything you can do using the RS-232 commands. We will need to essentially rebuild this software in python but also include a means to operate all 7 mass flow controllers. If we knew how to decompile this windows program and see how it was written that would make our job a lot easier. This is currently the best solution for interfacing with the OTC.

A screenshot of the main screen of the OTC software can be seen below.

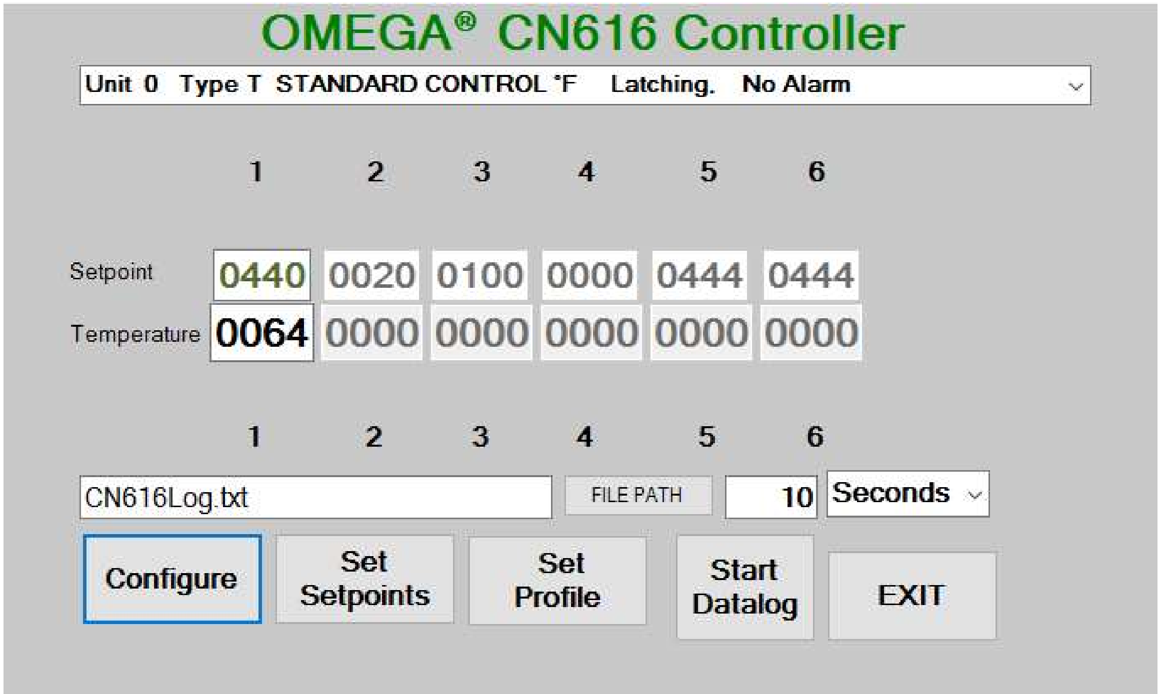

Figure 1: OMEGA CN616 Controller UI

For interfacing with the MFC, we are reading and setting an analog voltage. This is commonly accomplished through ADC and DACs. There is some documentation online about using the exact MFCs in the setup. They specify that an analog voltage needs to be supplied, it does not talk about which way this is accomplished, leaving this decision up to the user [1]. The current system is using a trimpot resistor as a voltage divider to calibrate 0-5V. An ADC/DAC will be able to function the same as the current setup but allow for digital control.

2.5 Proposed Design

Our proposed design uses an Arduino Microcontroller to act as a middleman between a user interface and both the temperature controllers and the mass flow controllers. This allows for modularization of the project to a degree. A change in a type of hardware only requires a few changes to the code. We have created a block diagram to show the solution we will be implementing; this can be seen below. The diagram implies a system with four distinct parts: the temperature controllers and their transceivers, the mass flow controllers and their DACs, the user interface and controls, and finally the microcontroller to connect the different parts together to allow for a common protocol between the different parts.

Below is the the overall block diagram of the system. This blocks represent the main functional aspects of the system, like controlling the gas flow or displaying information. The arrows represent the interfaces we are using to connect the devices together, what type of information is being sent. This system accomplishes the goal by integrating hardware, software, communication specifications, and design requirements.

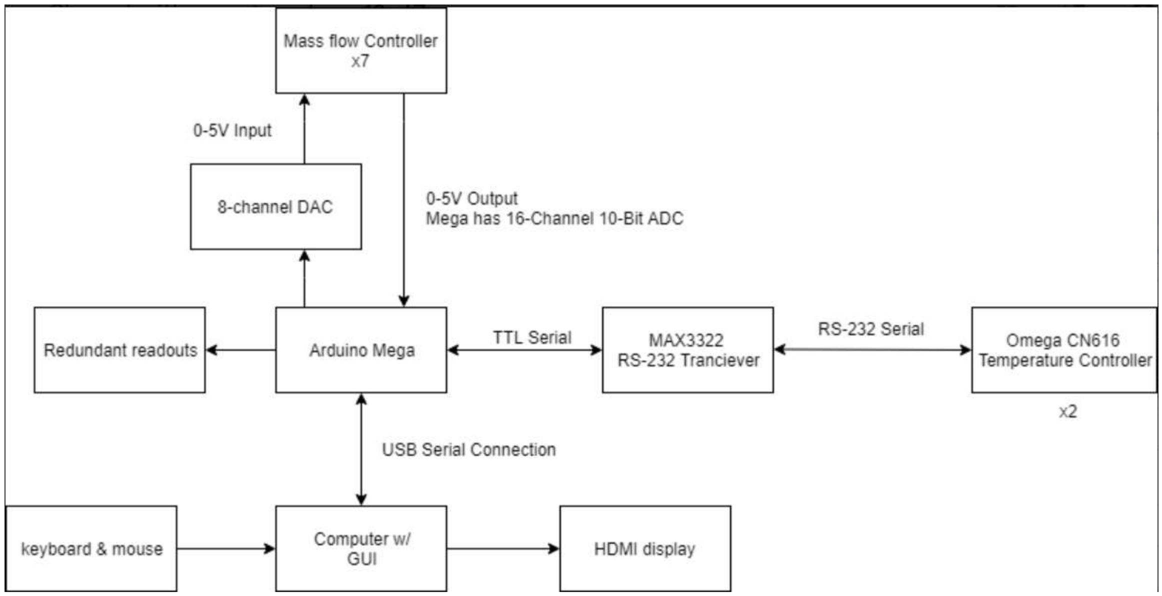

Figure 2: Block diagram of system

There have been some other considerations as to the flow of the project. We also considered using just a computer for the user interface of the system. This would eliminate the need for extra connections and components. Such a solution would also allow us to make a more comprehensive user interface that would remove the need for extra inputs for the mass flow controllers. However, we would need to do some more research to determine if such a solution would be economically feasible, so it could be something we consider in the future. In addition, we have considered several different microcontroller options, but chose the Arduino Mega because of its numerous connections and processing power.

Computer

The computer will be used to display the current readings of temperatures and gas flows. There will also be a user interface that allows for keyboard and mouse (or some other control scheme) entry. This will be sent to a display over HDMI. There will be a serial USB connection between the Mega and computer to allow the GUI to run the Mega as a slave.

Arduino Mega

The arduino Mega will be connected to the two OTCs over serial, relaying commands from the RPi to Max3323 to be converted to the RS232 standard. The mega will also have the key function of doing analog voltage reads off the 7 MFCs. It will also be communicating with the DAC to control the MFCs via a SPI interface. The Mega also will be taking in any analog user input including buttons and analog switches. Code will developed for the Mega using the Arduino IDE. We will need to use some more advanced features when we start dealing with the DAC, as this will not be using the 2-channel DAC present on the arduino. Analog reads will be also done, but the ADC for this is quite easy to use via Analogread().

LTC 1660 DAC

The DAC block performs the function of setting the flow rate for the 7 MFCs. This DAC will be soldered to the PCB shield which is in turn will be wired to the appropriate pins on the Mega: this includes pins for the data, clock, latch-in, and reset.

RS232 Transceiver

This block will connect the arduino Mega's serial to the RS232 level serial of the OTC. It will be soldered to the PCB shield and then connected to the TXRX of the Mega. There will also be some capacitors soldered around this to satisfy the timing requirements of the transceiver..

Arduino PCB Shield

Further along in our project after we have completed the prototyping we will develop a Arduino shield PCB. This PCB will contain the two RS232 transceiver chip and the 8-channel DAC. It will also contain the needed connectors to hook on top of the Mega and connect the MFCs and OTCs. This shield will be created using Eagle PCB software. The parts we will be using are readily available on ultralibrarian and snapeda. The design of the PCB will follow the following process:

1. Schematic creation- wire the circuit with components
2. place components footprints on PCB
3. route the traces

We will be assembling the PCB with parts found online using surface mount components

Software

The software for our project will be coded using python and the arduino IDE. The python element of our project will be using Tkinter for creating a GUI similar to the one made by Omega. It needs to effectively show the various states of the furnaces, allow for changing settings, change temperatures and gas flow rates, and prevent the user from entering something erroneously and causing any kind of dangerous situations. The layout of the GUI's main window will probably look very similar to Omega's. However, we expect to not include some of the extra things like data logging unless our design specifications change. We also are planning on adding other windows specifically for changing the temperatures and gas flow rates.

The Arduino Mega Software will contain code to transform the users commands into the appropriate calibration of the MFCs. This will allow a user to calibrate the device and then walk away knowing that the gas flow will be adjusted at the desired time. This functionality can be seen in the figure below.

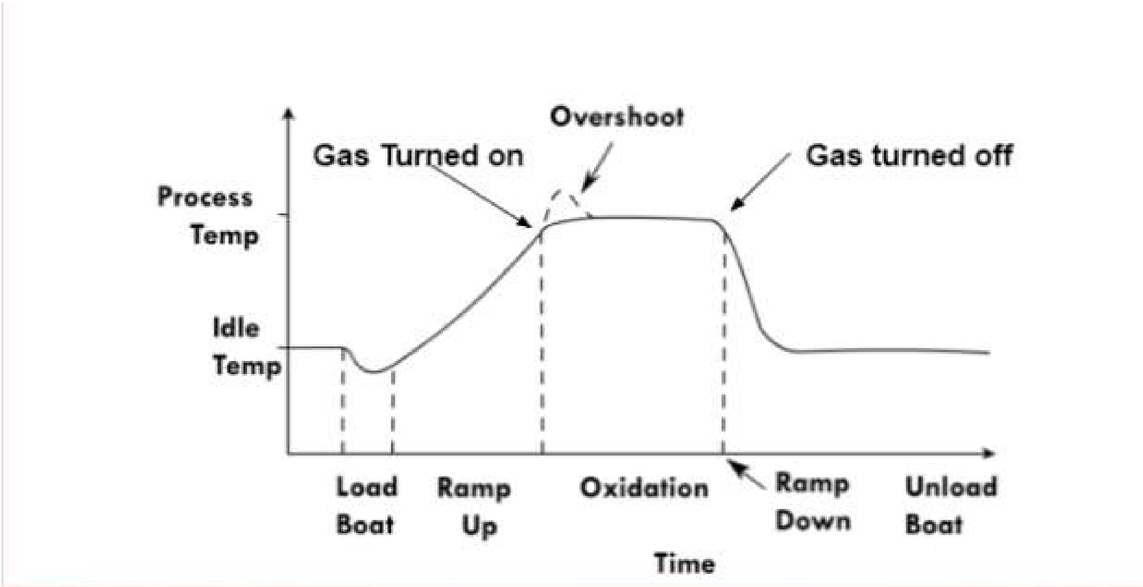

Figure 3: Temperature profile

Our software will keep track of the time to ensure that the gases are turned on and off at the desired points required by the user. This time-temperature data series that goes with each profile will also be able to be saved to a datalog for the user to perform analysis with later. This is helpful because then with this data they can compare theoretical expectations to actual results for various oxidation and doping process'.

2.6 Technology Considerations

Fortunately most of the technology used in our design is readily available and within our budget. Much of the hardware we have and need is also available from our faculty contact.

There are several pieces to the project, and we had considered possible simplifying it down to using only a microcontroller or just a computer. However, there are a lot of built-in features present in one system, but not in the other. For instance, the computer can easily run a Python GUI on a screen with any kind of graphics card or special considerations, but there is no simple way of attempting RS232 communications with it. The microcontroller we are using can do RS232 communication, but has no simple way of implementing a GUI. Combining the two pieces of hardware only costs a little extra, but saves many hours of work time.

Depending on which serial interface we choose to go with for the DAC, we will need to ensure that there is appropriate hardware setup for this interface. With I2C this means that we have appropriately sized pull-up resistors for clock and data lines. With SPI there is similar hardware requirements. We will need to carefully read up on the datasheet for whatever DAC we choose to ensure we are correctly wiring the pins.

Besides the DAC, we must also ensure that we are following the correct wiring guidelines for the RS232 transceiver. This entails making sure we have the correct sized capacitors hooked up to allow for timing guidelines to be met.

In investigating which DAC we should choose we read into the MFC datasheet. The data sheet says that mass flow is accurate to within 1% of the applied voltage. This means that our DAC should be at least 1% accurate. There is a common formula that can be found in literature, the accuracy of a DAC is equal to ½^n [2], so choosing an 10 bit dac will give us around 0.1% accuracy, which is enough for the application.

The next aspect of the DAC we researched was the interface. There are 7 MFC's in our system which means that the number of ports an 8-channel (there are no 7-channel DACs) 10-bit DAC would need is 64, too many to deal with. Thankfully there exists serial DACs (as opposed to parallel) that can be programmed via I2C and SPI. The arduino has both I2C and SPI so deciding between these two kinds was the first choice. We found that for our application we could go either way.

Our dac would be on a PCB above the arduino (a shield) so the distance was not long enough to be rule out the SPI interface. The data transmission rates was going to be pretty small so I2C would also be suitable. In the end we would be going with whatever was most readily available online. We found that most of the serial DACs use SPI so we went with that [3].

2.7 Safety Considerations

Safety considerations should always be focused toward protecting people. This project requires very expensive and specialized equipment however, so we will also discuss safety considerations in regards to protecting the equipment.

Human Safety: As with any electronic device, caution should always be used around frayed or exposed wires. Our design plans for a tidy circuit, however, under heavy usage, wear and tear can occur. Additionally, caution is important when connecting the control system to the furnace.

Machine Safety: It is important to have a good understanding of all wires and cables when connecting the control system. An incorrect connection could damage both the control system or the furnace.

Due to the fact that our PCB will be accessible to users, we may need to incorporate ESD protection to protect the communication lines. This is something we will need to investigate further once we know the exact lengths our data is traveling and what protocol, whether RS232 or just analog voltage.

2.8 Task Approach

When approaching this task we find it important to use Design Thinking. We created a flowchart to showcase our design process. We find it important to start by properly defining the the issue that we are solving. Then we use that defined issue to brainstorm solutions. The brainstorming process is very important and is used continuously throughout the project, not just in the beginning. After we have settled on a solution we will then deploy our resources and set tasks for each team member. The final goal of these tasks is to create a prototype that can then be tested. If the prototype has issues then brainstorming will be used again to try and fix the issues. Finally, when we are comfortable with our prototype, we will test our final design among users and get any final feedback. The overall design process can be seen in Figure 1.

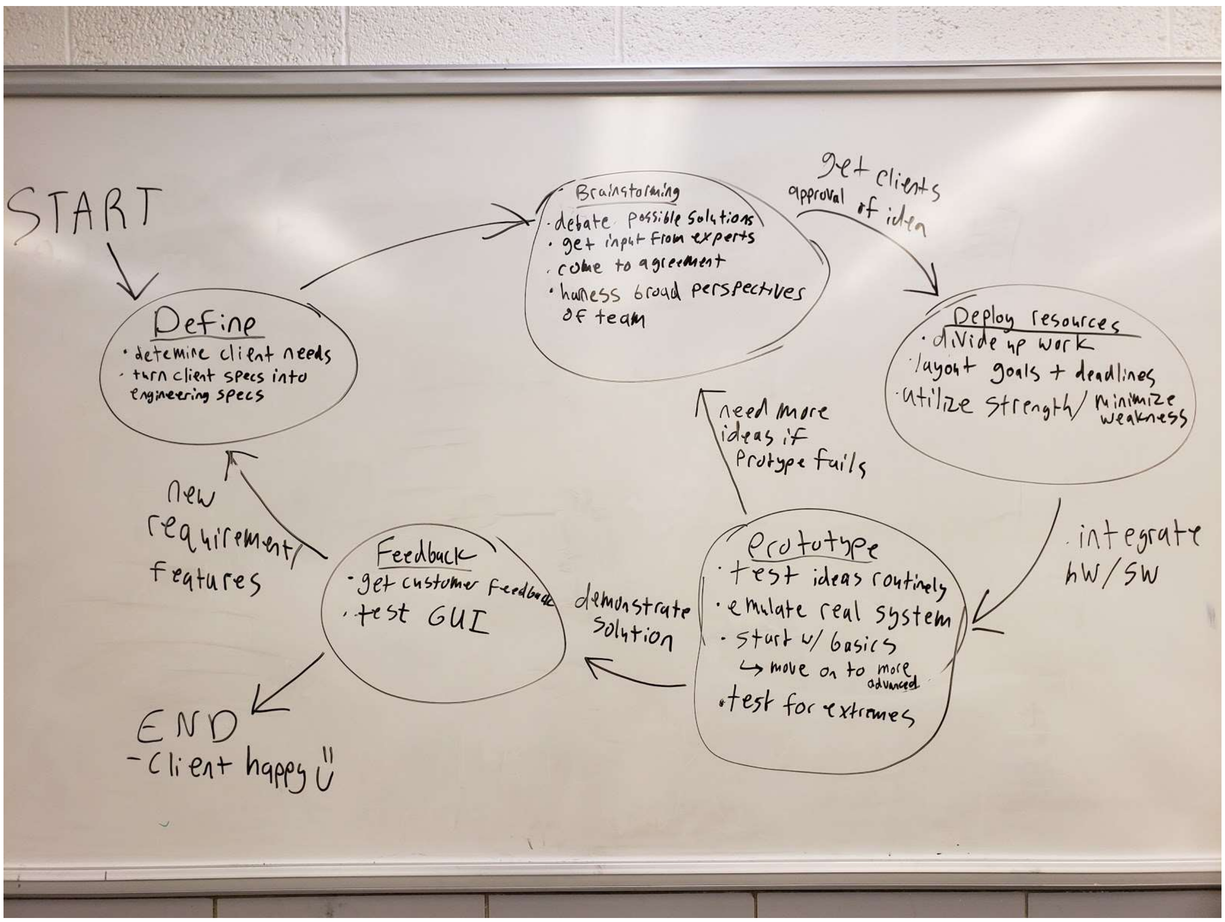

Figure 4: Proposed Design Diagram

The overall task approach will be as follows.

- Nick is in charge of the system engineering and verifying that the components that we have chosen will work. He will be in charge of making sure the software and hardware are being correctly interfaced. Jeremy will also be involved heavily here, helping to make sure the software developers understand the functional requirements of the system.

- Adam and Chris will be the lead software developers. Chris will be in charge of developing the python GUI that the user will use to send commands. Adam will work primarily with the arduino code, developing the timing control of the system while also coding in redundancies/error checking. He will make sure the arduino is correctly interpreting the commands.
- Kevin and Jeremy will in charge of design the prototype and PCB; doing most of the hardware debugging and testing. They will be responsible for the procurement of the hardware necessary for the system.

2.9 Possible Risks And Risk Management

Some issues that might arise in the testing and planning stages of our project are:

- Technical/area of knowledge limits. For instance, one problem is if we cannot get the furnace to communicate with the microcontroller we have to planned on using things will not work.
- Physical limitations. If some of the parts we buy are faulty or even are delayed. Also if the parts we need or have planned becoming greater than the budget we initially planned on there will have to be changes in our design and future plans.
- ordering a PCB - there is always the chance that the traces are screwed up somehow. We will need to make sure that the board is good when we send it in to the fab house.
- Another risk is that the microcontroller on the OTC exhibiting unpredictable behavior that our code will need to account for. We will need to incorporate error checking to make sure the microcontroller sends back something that we cannot interpret.

2.10 Project Proposed Milestones and Evaluation Criteria

Event	Criteria
Send commands over serial to the OTC	Being able to set/read temperature, perform commands
finish python GUI	be able to set/read temps and change OTC config, send change temp commands to arduino
Get 8-channel serial DAC working with arduino	verify with breadboard and multimeter that arduino can send serial (SPI) data to change voltages on each of the 8 channels

integrate MFC calibration into existing OTC temperature profiles settings on GUI	test full setup with multimeter. verify that MFC flow is changing at specific time interval
Send finished PCB gerbers off to board house	ensure all traces have been routed and schematic is correct
solder components onto PCB	do rigourous circuit testing to make sure the wiring is correct we will also need to make sure to find appropriate connectors so all wires can be safely secured
install our solution into the NSF lab and wire everything	extensive testing with real temperature profiles to make sure the system can run long time

Table 1: Milestones

2.11 Project Tracking Procedures

To track our progress we have weekly status reports. These reports help us keep track of what we accomplished, what issues occured, our goals for the next reporting period, and the individual contributions of each team member.

We are also using Gitlab's built in project management system to keep track of work we need to do. We can create issues on Gitlab telling us when the work we need to do is getting done and for making it easy to find resources we have found and are using. After each revision the code we are working on will be pushed to gitlab for both the python GUI and arduino code.

2.12 Expected Results and Validation

The expected results of this project will be fully operational software and hardware that allows the user to view the readouts of the system. It will also allow the user to directly calibrate the system, or set up a temperature profile for the system to change the values over a set period of time. The user will have the option to output the time series data of the temperature and gas flows to a file for their own analysis. This software will be in the form of a python program. The hardware will be in the form of a PCB shield fort the Arduino Mega containing the RS232 transceiver IC, DAC, and connectors for the MFCs and OTCs.

The system will be validated in the NSF lab by connecting the 7 mass flow controllers and two temperature controllers onto our Arduino Mega Shield via appropriate wires and connectors. The Arduino Mega will be connected via USB to a computer running our python GUI. We will first validate that temperature setpoints are working for each of the 2 furnaces 3 zones. This is a RS232 command that can be validated by manually checking the setpoints on the OTCs readout. This is the same for the current temperature value, verifying that the value we receive in the GUI is the same as what is on the OTC.

The mass flow controllers will be validated next. This will be accomplished by the sending out the 0-5V signal from the DAC to each the MFC followed by probing the signal at the MFC input to verify that the correct voltage is being applied. Verifying that the correct gas flow rate is occuring depends on the factory calibration of the device.

2.13 Test Plan

Our testing plan starts with testing the communication between the Omega Temperature controller and the Arduino. This test involves testing two different RS232 adapters. One is an RS232 shield for the Arduino, the other is a MAX3223 chip. Testing these two interfaces will determine if they allow satisfactory communication between the arduino and the OMega Temperature controller. Additionally, it will help us determine which method is best for our project. We will do this by sending read and write commands to the omega controller and see how it responds.

The communication with the Mass Flow Controllers and the Arduino will also require testing. We have ordered several different 8-channel digital to analog converters. Using these chips, we will test the accuracy and results of communication with each one. The results of this test will allow us to decide which one to use, as well as help us debug any communication errors to the MFC. It is very important to have accurate communication to such a sensitive instrument. We will perform testing with the arduino and a multimeter to verify that the serial data sent to the DAC is correctly setting the outputs of ours DAC.

Furthermore, we will need to, after testing basic communications, test the communication through the GUI. We will run a full series of tests that verify different aspects of the GUI and the overall control system. These tests will look for time delays and misinputs, in particular. The GUI will also need extensive testing to ensure that it easily navigable by the user, do the navigation displays make sense for the system we are trying to implement.

A large chunk of testing will be to verify that the PCB we have ordered for the arduino mega. This will be done after the components have been soldered onto the board. It will include verifying that the DAC output can be easily set by the arduino. We will also verify that the RS232 transceivers are indeed operating and can interface the OTC.

After thorough prototype testing of our control system, we will then test it with the NSF Furnace and mass flow controllers at the MRC. Having the whole system together will allow us to search for any final, unexpected errors.

Finally, we will test our user interface with researchers and students. We will then ask for feedback and take any user suggestions. This will be our final test before we decide to release our final product.

3 Project Timeline, Estimated Resources, and Challenges

3.1 Project Timeline

Our overall project is planned to span two whole semesters at Iowa State University. This breaks our timeline down into two main sections. The first section/semester is from August to December and is operated through EE 491. Our project will then continue from January to May through the course of EE 492. These two sections of our project timeline can be visualized by the black vertical line in our GANTT chart (Figure 3).

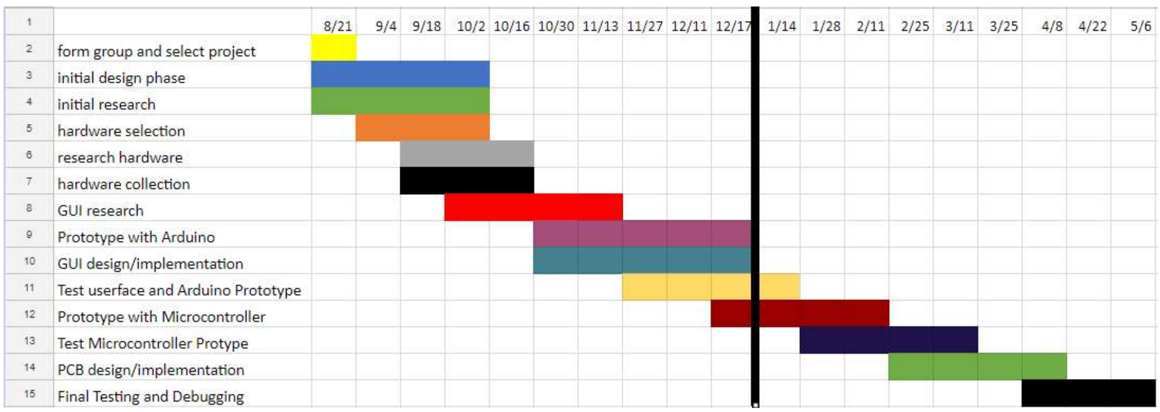

Figure 5: GANTT Chart

Our timeline for the first semester of our project (August-December) covers a lot of initial research, design, material selection, and planning. We set aside the first month after forming our group to work on the initial research and designing for how we will solve the problem of creating a NSF Furnace Control system. Time researching was not only spent online and in books, but also by interviewing Dr. Tuttle and assessing the current NSF Furnace control system.

After we felt comfortable with our project and what we wanted to accomplish, we were able to create a more detailed and precise project plan. This timeline provides a lot of overlap between each section and goal. That allows us to adapt our timeline in the case that we finish one area sooner than expected or later than expected. Because we are all still currently students, our schedules can vary. One week we might have a lot of exams and then have a little bit of an easier week afterwards. This GANTT chart allows us to adapt to our ever-changing class loads.

We also provided a lot of time for prototyping and testing. These two sections are the largest portions of our project (and most projects). During this time we will be able to implement our research and designs and then verify. With this large amount of dedicated time we will be able solve any unforeseen bugs to ensure our system works completely by the end of our project timeline.

The second semester of the project will include the majority of the technical improvements of the project. The first semester's main focus of research and basic prototyping will allow for extensive technical work in the second semester. Major prototyping will likely continue into second semester and will lead into a final production phase by the middle of the semester. By the end we should hopefully be able to implement our project into the furnaces in the NSF lab for final testing.

3.2 Feasibility Assessment

One of the main challenges we will face is ensuring that our solution can be successfully integrated into the system. We will be removing all of the current manual controls for the furnace leaving only digital controls. We do not have a schematic (our client does not have one to give us) for the current system (the MFC wiring), so we are unsure of what we will be removing and what will be staying put. Our biggest question is still the power supplies for the MFCs. Without schematics or specifications we don't have a way of confidently integrating them into our system. We understand how to power the MFCs given by the datasheet specifications so we should be able to keep the current power supply. The current manual controls for MFC can be completely scrapped as we are implementing a digital solution.

The communication protocols used with this project are not complicated. RS232 consist of commands that have been written out in the OTC datasheet. For controlling the MFCs, we simply need to set/read analog voltage values. For these reasons we have chosen a mix of hardware, taking advantage of the arduino integrated ADCs while adding an external DAC. For these reasons we believe the hardware aspect of our system to be very feasible. Overall, the devices we are using have excellent documentation that should make debugging unforscene problems trivial.

The software of our system will be a more challenging aspect. We have many functionalities that we want to add to this system. We will need to find a way to integrate the existing commands that are able to sent to the OTC and add the MFC into the temperature profiles to achieve true user automation. It would be quite simple to allow the user to set/read temperatures and gas flow rates manually over a digital interface. The challenge will be allowing the user to truly automate this process, ensuring that they can walk out of the lab and come back when it's done. This is software that exists, it is already present internally to the OTC, we just need to make our own that also incorporates the MFCs.

Our project ties different systems together under one controller and user interface. This has the potential to cause some problems. Different parts of our system require different methods of communication, therefore we require DAC and ADC chips. All of the different communication interfaces can cause trouble and be confusing. That is why it is important to test communication with each individual control system before we put them all together under one controls user interface.

Additionally, further challenges can be mitigated by deciding on the proper equipment to communicate with our controllers. We have done this by deciding to use both a computer and an Arduino. These two products will work hand-in-hand to properly communicate with the MFC and OMEGA temperature controller. The Arduino has more serial ports and a 5v output that works well with the Mass Flow Controllers.

3.3 Personnel Effort Requirements

Task	Description	Estimated Time
Python GUI	display readouts & calibrate system	30 hours
MFC control code	arduino Code to set/read mass flow controllers	20 hours
OTC control code	arduino C code to interface with OTC	20 hours
Hardware prototyping	Ensuring the DAC & RS232 IC's are operating properly	20 hours
Arduino PCB shield	• designing PCB to house DAC and RS232 ic's • also includes soldering on parts after it arrives	20 hours
installation	wiring up all hardware into the furnace control system	8 hours

Table 2: Personal Effort Requirements

3.4 Other Resource Requirements

This project will naturally require several different components and parts. A comprehensive list of basic parts that we will need to acquire (that aren't naturally already in our possession). We already had an Arduino Uno given to us by Dr. Tuttle. However, for our full project, the Uno does not have enough serial ports required to communicate with all of the Mass Flow controllers. Therefore, we will need to purchase an Arduino Mega. Additionally, a numpad will be needed for users to input temperature values into our control system. A screen will also need to display values and information to users. Additionally, an 8-channel DAC will be needed for the Arduino to communicate with the mass flow controllers.

Parts/materials:

- Arduino Mega
- Numpad
- Screen
- DAC
- RS232 converter

3.5 Financial Requirements

The finances for this project come from Iowa State University's Microelectronic Research Center. Our use of these funds are controlled and approved by Dr. Tuttle. We have been given a loose budget of $500-$700. However, with our current project plan, our overall expenditures will be drastically below budget. The majority of our money will go towards and Arduino Mega and screen for our user interface. We estimate a total expenditure less than $300. Below is our current bill of materials and estimated prices.

Part	cost
Rpi Model 3	$35
Arduino Mega	$40
Arduino Shield PCB	$100
ICs (RS232 TXRXand DAC)	free samples from TI and ADI
PCB connectors	$20
Display screen	$100

Table 3: Bill of materials

4 Closure Materials

4.1 Conclusion

This project will be upgrading the electronics and making the user interface completely digital. The upgraded solution will make the furnace much easier to program and use by automating the ramping of temperatures and gas flow. We will be accomplishing this by sending digital commands over RS-232 and ADC/DACs, as opposed to the complete manual input that is currently used. This will allow lab TA's and users to set the parameters and not have to worry about coming back into the lab to adjust temperature and gas flow rates, it will be an automated process.

We will know if we succeed if we can successfully get the hardware integrated into the system and run some wafer temperature profiles. This will make the job of TA's in EE 432 much easier as they will be able to leave the room to go work on more important research, they will be able to run profiles overnight, and come back in the morning to some nicely doped wafers.

Some of the biggest challenges facing this project will be software development and being able to prototype our solution well enough before the final installation. If we are able to overcome these issues we will succeed. This will come from careful reading of documentation and a looking to other GUI control systems for inspiration.

4.2 References

1. "Tylan FC-2900/FM-3900 Mass Flow Controllers and Flowmeters" *http://www.innovactech.co.kr* [Online] Available: http://www.innovactech.co.kr/upload/catalogue_file/e59cd23707c808e69834533440aded96.pdf [Accessed October 31, 2018].
2. "RF Wireless World," *rfwireless-world.com*, 2012. [Online] Available: http://www.rfwireless-world.com/calculators/n-bit-DAC-resolution-calculator.html. [Accessed October 31, 2018].
3. "SPI versus I2C protocols" *bitwizard.nl*, July 18, 2014. [Online] Available: https://bitwizard.nl/wiki/SPI_versus_I2C_protocols. [Accessed October 31, 2018].

4.3 Appendices

4.3.1 Temperature Controller

4.3.2 Mass flow controls