SOFTWARE-BASED AUTOMATION SYSTEMS FOR CRYSTAL GROWING PROCESS

BY

JOHN HORNIK
RESEARCH IN COMPUTER SCIENCE
FINAL PROJECT
LEWIS UNIVERSITY
ROMEVILLE, IL, USA
MAY 2017

TABLE OF CONTENTS

List of Figures

Abstract

This project focuses on the development of software-based systems to assist in controlling the crystal growing process at Siemens Medical Solutions in Hoffman Estates, IL. More specifically, it focuses on inexpensive, yet sophisticated manners of updating the crystal growing furnaces, the crystal cutting saw, and the environmental control in the crystal dry room. The main goal of this paper is to present ways in which the field of computer science can help solve some of the issues associated with the control systems within the Crystal Growth Department. It should be noted that the current systems were developed decades ago and are slowly beginning to fail. The sodium iodide in the air of the department also plays a role in this as it is extremely corrosive and can cause damage to many moving parts. Implementing newer computerized controls would omit many of these problems as integrated circuits have been proven to last longer in this environment. Additionally, because these systems would be programmed, there would be much more flexibility within their design and can be molded to fit the exact preferences needed to continue for many years to come.

New software-based concepts for the process control system revolve around implementing machine learning techniques into a few of the sub-systems of the furnaces. For the string saw, which is used for cutting the crystal blocks into thinner pieces, an embedded system is presented which utilizes a microcontroller in line with the current sensors. Finally, for the environmental control, a microcontroller-based HVAC design is presented which may prove to be cheaper and more efficient than the current system. The paper concludes with a brief summary as well as a section on automated theorem proving. It also covers some of the dangers that automation brings to job security. All code examples within the paper are based on the C programming language which is implemented with the Arduino IDE.

Chapter 1 – An Introduction

1.1 Crystal Growth Department

The Crystal Growth Department for Detectors within Siemens Medical Solutions grows sodium iodide NaI(TI) scintillation crystals for use with SPECT scanners. The crystals begin as raw salt material that are brought through a number of steps, or stages, including drying, heating, and cooling down. The raw material is then placed into a furnace for the actual growth process. Here, it is heated to extreme temperatures until crystallization occurs and the salt forms into a solid ingot, or crystal block. After this crystallization occurs, the ingot needs to be cut, pressed, and shaped before it's ready to be formed into a plate size, framed, and installed in a scanning device.

Currently, the control systems for this process are based on custom designs comprised of mainly analog electronics. Essentially, the ideas presented deal with using digital computers to solve the process control equations rather than the current analog circuits. This occurs with the analog signals of given transducers being converted to binary data. Software algorithms can be implemented in order to handle all operations, such as addition, subtraction, division, multiplication, etc. These algorithms can then be used as necessary in order to solve future process equations [1].

1.2 Motivation

Many of these processes are automated to some degree and many of them possess digital controllers as a form of control. As an example, one of the furnace control boxes can be seen in Figure 1.

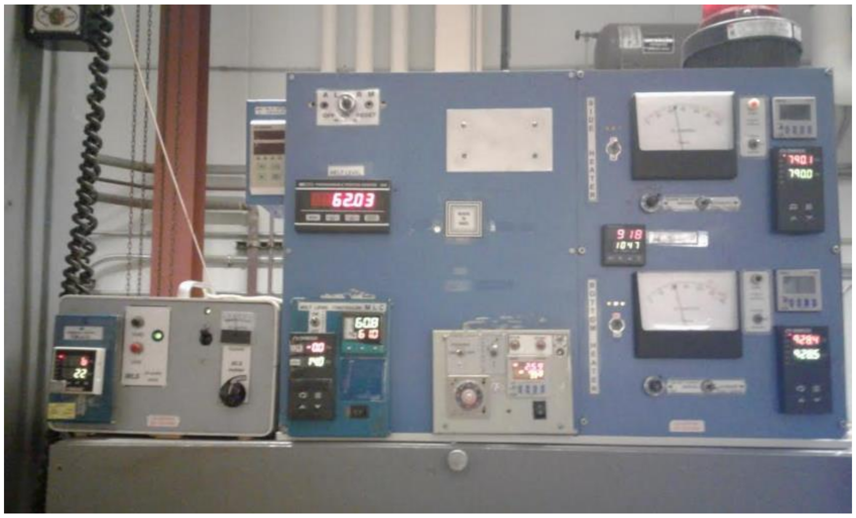


Figure 1
Control Box for Furnace Control

With the current advances in technology, one of the main goals is to expand the use of embedded control systems within the department in order to replace some of the older analog controls as well as all of the more expensive off-the-shelf digital controllers. The idea originated primarily due to the consistent failures of the current systems in place. Some of these systems are approaching 30-40 years and are in dire need of replacement.

One of the biggest issues with moving forward is that some of the current electronics are extremely outdated and repairing or replacing the components within them would only extend their lives by another year or two.

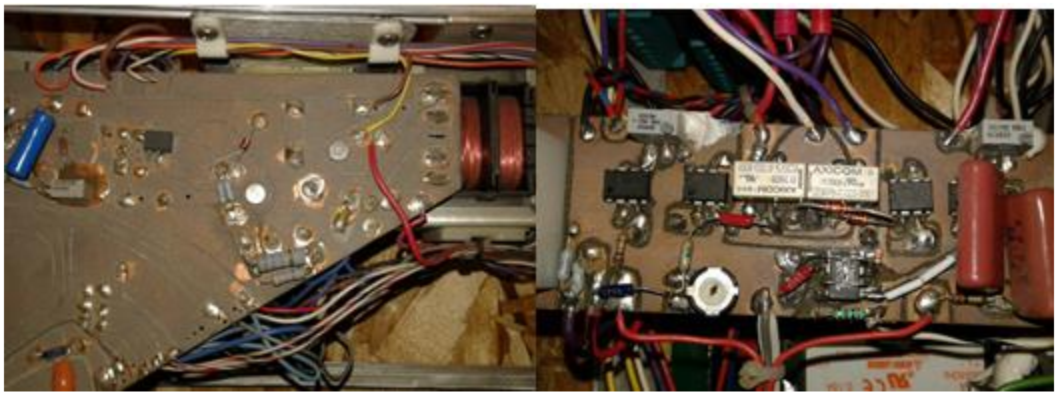


Figure 2
Examples of Current Analog Circuitry

An example of the outdated electronics can be seen in Figure 2. Because of this, the paper presents a means to redesign and implement new systems which are mainly digital and controlled by software rather than analog logic.

1.3 Project Review

This project is organized into seven chapters. The first chapter presents an introduction to the project as well as the motivation for the project. In Chapter 2, the current process control system is presented followed by the proposed system which implements a more intelligent software-based solution. Chapter 3 covers the current crystal cutting string saw system. This chapter also concludes with a proposed system which is more digital in nature. In Chapter 4, there is a presentation of a wireless temperature control system which can be implemented into a dry room which requires accurate temperature control. Finally, the paper concludes with a summary of the project as well as the importance of theorem proving and some negative effects which may arise from implementing digital automation into the Crystal Growth Department of Siemens Medical Solutions. Chapter 6 contains the bibliography while Chapter 7 displays sample code for the new string saw system. It should be noted that all designs are strictly theoretical and based on the research done for this project. Similarly, all code and designs based on the code are proof-of-concept and are not ready to be implemented. They were also created based on ideas generated through this project and require much more software and hardware as well as verification to be sufficient for everyday use. Because this paper focuses on software and not hardware, there are very few references to external hardware components, such as relays, transistors, transducers, etc., that would be necessary in order for these designs to be properly implemented for control.

Chapter 2 – Crystal Growth Furnaces

2.1 NaI(Tl) Crystals

The crystals grown at Siemens Medical Solutions are used in radiation detection devices. More specifically, they are used in single photon emission computed tomography (SPECT) and positron emission tomography (PET) devices in helping locate and diagnose a number of diseases. The techniques involved include computed tomography (CT) as well as a radioactive material (or tracer). The tracer is essentially what the doctor studies in order determine blood flow to tissues and organs. Once the tracer is injected, it emits gamma rays and these are detected by the scanner. The computer within the scanner then takes this data and creates cross-sections of whichever part of the body needs to be focused on [2]. The crystals themselves are installed in thin metal frames and are encompassed in a glass casing. They are mounted within the gamma camera (which captures the gamma rays) and work in conjunction with photomultiplier tubes (PMTs) in order to capture the images.

2.2 Crystal Growing Method

The crystal growing process that is implemented by Siemens is similar to the Czochralski Process. This method is used to obtain single crystals of sodium iodide doped with thallium. It begins with a seed crystal (or cylindrical piece of previously grown crystal) being mounted on a shaft within the furnace. An example of the furnaces can be seen in Figure 3.

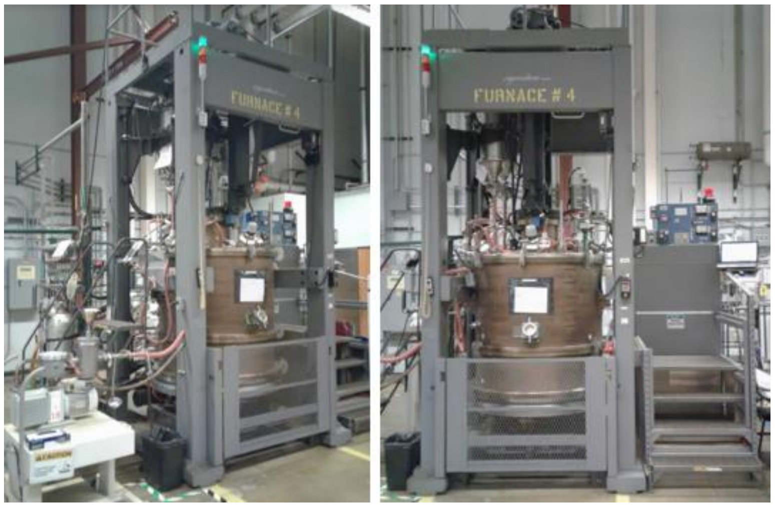


Figure 3
Crystal Growth Furnace

The atmosphere within the furnace is typically near 1000°C in a vacuum, pumped out with Argon gas. The seed is then dipped into a melt of molten raw sodium iodide material. The seed's shaft is pulled upwards and rotated at the same time. By precisely controlling the temperature gradients, the rate of the pull, and the speed on rotation, it is possible to extract a large, single-crystal, cylindrical ingot from the melt [3]. Figure 4 shows a basic example of how this process functions.

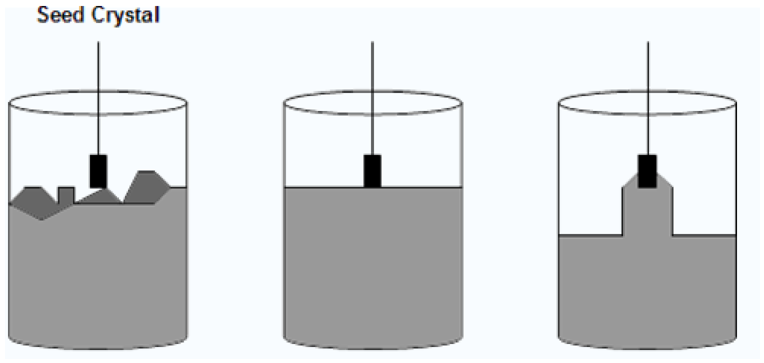


Figure 4
Czochralski Method

2.3 Current Process Control System

The furnaces are composed of a number of different systems. The following are the main subsystems of each furnace:

- Crystal Rotary Axis Servo Control: The Crystal Rotary Axis Servo Control controls the rotation of the crystal holder. The servo control system maintains a fixed RPM that is in proportion to the speed of the crucible.
- Crucible Rotary Axis Servo Control: The Crucible Rotary Axis Servo Control controls the rotation of the crucible. The servo control system maintains a fixed RPM that is in proportion to the speed of the crystal.
- Elevator Lift Axis Servo Control: The Elevator Lift Axis Servo Control controls the position of the lift that contains the crystal holder. In addition to the encoder that is attached to the servo motor, two additional encoders are provided to monitor the position of the holder and to ensure that each of the screws are turning the same distance.
- Level Detection Stepper Control System: The Level Detection Stepper Control System consists of a stepper motor and controller that raises and lowers the platinum probe that detects the position of the molten bath. The position of the probe, when in contact with the bath, is used to determine the growth rate of the crystal. Limit switches are incorporated to limit travel.
- Heater Temperature Control System: The Heater Temperature Control System consists of two temperature control systems that control the amount of energy applied to the heater elements. Two separate controllers are present for the pre-melt and crucible zone. Each zone has two thermocouples in order to sense the temperature. These thermocouples are provided for redundancy so that if one should fail, the other unit will provide as a back-up. The system also allows for the entry of a desired power level should the operator wish to operate the system based on power input rather than temperature. Transformers are supplied for ground isolation.

- Feed System Control: The Feed System consists of two feeding cylinders that house the raw material which is used in the bath. Electric winches are used to raise and lower the cylinders.
- Cart Control System: The Cart Control System consists of two motors with variable frequency drives that control the horizontal and vertical positions of the cart. Limit switches are implemented to monitor the position of the cart.

Most of these systems are currently controlled by the analog circuitry described earlier in the paper as well as with off-the-shelf controllers. A controller is a device which receives information about a process, makes a decision based on this information, performs some action on the process, and monitors the result [1]. There are many controllers (process, temperature, limit, etc.) available on the market; however, many of them cost hundreds to thousands of dollars. Because this paper focuses on cost-efficient designs, we will emphasis on means to create our own for the given process. The basic layout of a digital controller is shown in Figure 5.

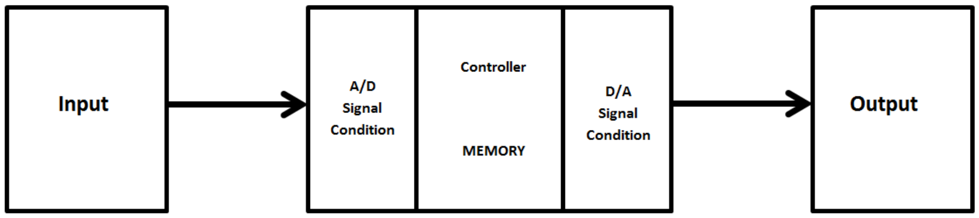


Figure 5
Digital Controller

The input can anything (switch, light, transducer, etc.) that sends an input signal to the controller. The controller then processes the signal from analog to digital, performs the actions that it was programmed to, and converts the information from digital back into analog. From there, the output signal gets sent to motors, solenoids, relays, displays, or other peripherals that are being controlled.

2.4 Proposed Process Control System

In order to improve the deficiencies within the current system, a software-based approach would be the most effective. Implementing a series of digital IO devices, such as microcontrollers, would allow for many of the systems to operate autonomously. An example would be the Heater Temperature Control System. In many cases, the operators, which continuously monitor the system, constantly need to make split-second decisions when an issue occurs. One notable problem which arises quite frequently would be fluctuations in temperature conditions. Mainly due to spikes in heater current, the operators need to quickly raise or lower the crystal block from the melt below in seconds. If they are unsuccessful, damage may result to the ingot. Implementing a Machine Learning method would alleviate this problem. The K-Nearest Neighbor (KNN) algorithm is one which is based on supervised learning, or learning in which a training set contains data and the correct output of the task with that data [4]. In this case, if an operator always raises or lowers the ingot under certain temperature conditions, the algorithm would follow these conditions. Based on these previous occurrences, the system would "learn" that the ingot automatically needs to be raised or lowered in future instances of these conditions. In this scenario, the KNN algorithm takes on a "classification problem". This type of problem is binary in nature as it has two forms, in this case when there are jumps in temperature, the system needs to raise or lower [4].

Another situation where the KNN algorithm would be useful is with the Feed System. When the levels of the hot melt raise or lower, the operator must make a decision whether or not to add more raw material from the hopper (or storage container). Again, based on variations in melt, the system would need to make a decision, feed or don't feed. Both of these examples are based on learning. In order to succeed, inputs (such as variation in temperature or melt level) need to be given to the system so that it can react appropriately. The information conveyed to the system is used to create and modify the knowledge structures within its knowledge base. This knowledge then creates necessary outputs based on what it has learned. One of the most important aspects of this system becomes its ability to receive feedback. With this, it is able to determine whether or not its performance was acceptable. With proper actions, the system's performance will have improved with the changes made to the knowledge base [5]. In this case, acting on either feeding or not, or lowering or not, would produce a result. Feedback informing the system whether or not its conditions have improved would be the determining response of its proper or improper operation. Such a system would omit issues that occur with furnace operation on a daily basis and would make the control easier and much more efficient as a result.

Chapter 3 – Crystal String Saw

3.1 Current String Saw Design

Prior technologies to the string saw include metal band saws as well as filament saws. Many issues arise when using these tools due to the sensitivity of the crystal material. The metal teeth on these saws would chip away at the material resulting in loss of product which in turn can cost the company thousands of dollars. The current system utilizes a water-based string saw. Generally, a cotton and polyester blended string is used which is run through water before contacting the crystal ingot. This allows for more of a clean slicing operation as the crystal almost melts from the water and friction of the string.

The current system employs two motors and two sensors to cut crystals. The first sensor is used to detect speed of the saw while the second motor senses the tension of the string. When the saw is turned on, it goes through a basic control process:

- Power button is pressed.
- The string motor and linear motion motor are both engaged.
- The linear motion speed of the saw is monitored by the first sensor.
- Any variations in speed are controlled by the first sensor.

- The second sensor senses the tension of the string.
- A variation of tension in the string is closely monitored and controlled by the second sensor.
- The filament is soaked in solvent.
- End.

Another important aspect of this process is a timer that runs from 0 to 100 seconds. The timer function is essential in protecting the integrity of the crystal in cases where the motor moving the saw fails and the motor turning the cutter continues. This can cause serious damage to the crystal and in many cases it would need to be discarded. The string saw can be seen below in Figure 6:

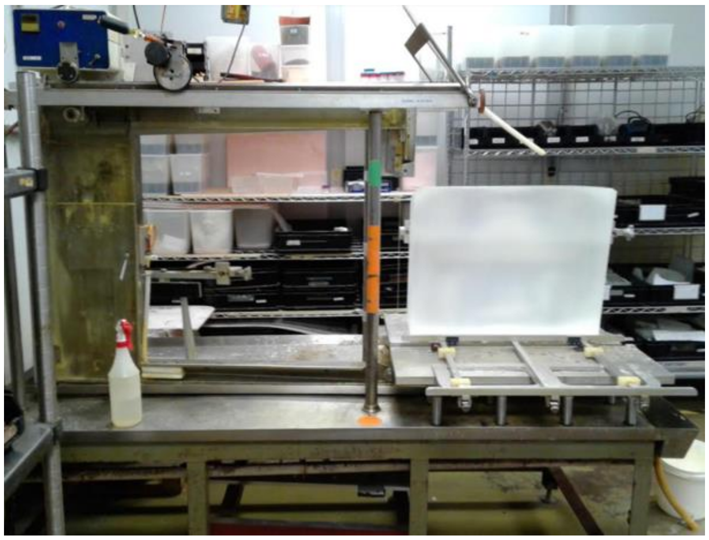
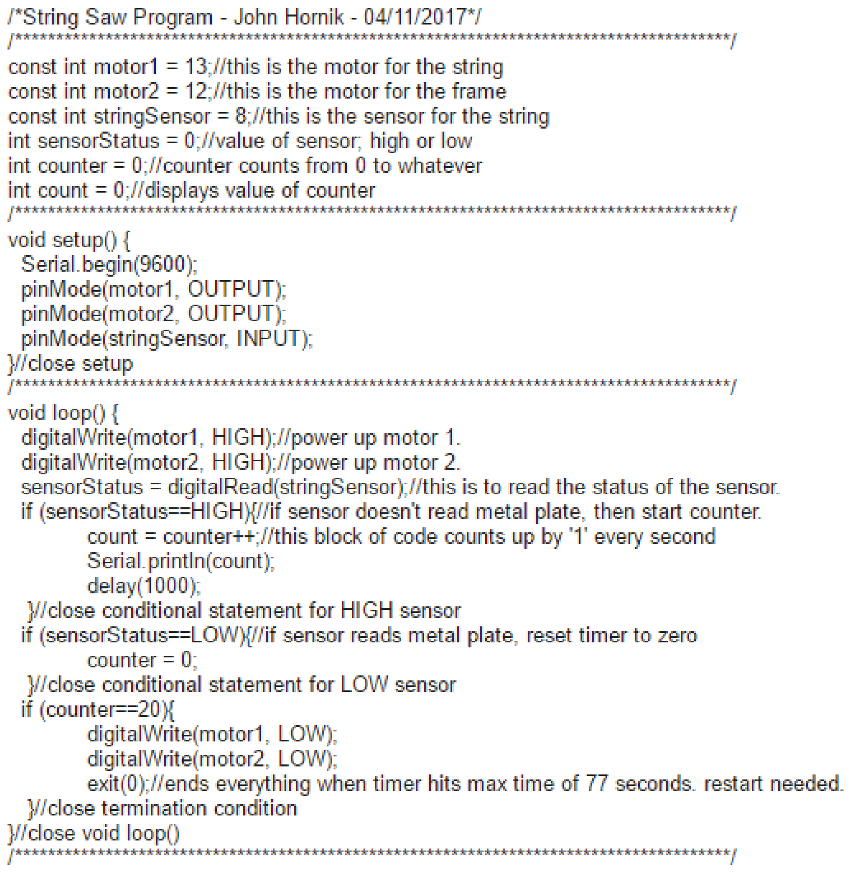


Figure 6
Crystal String Saw

3.2 Proposed String Saw Design

During the planning phase of this project, a digital version of the above system was replicating in code. Utilizing an Arduino microcontroller, a simpler design was developed. This design essentially mimics most of the functions of the analog hardware controls within the string saw's control box. The code that has been developed for this design can be seen in Figure 7.



```
/*String Saw Program - John Hornik - 04/11/2017*/
/**********************************************************************************/
const int motor1 = 13;//this is the motor for the string
const int motor2 = 12;//this is the motor for the frame
const int stringSensor = 8;//this is the sensor for the string
int sensorStatus = 0;//value of sensor; high or low
int counter = 0;//counter counts from 0 to whatever
int count = 0;//displays value of counter
/**********************************************************************************/
void setup() {
  Serial.begin(9600);
  pinMode(motor1, OUTPUT);
  pinMode(motor2, OUTPUT);
  pinMode(stringSensor, INPUT);
}//close setup
/**********************************************************************************/
void loop() {
  digitalWrite(motor1, HIGH);//power up motor 1.
  digitalWrite(motor2, HIGH);//power up motor 2.
  sensorStatus = digitalRead(stringSensor);//this is to read the status of the sensor.
  if (sensorStatus==HIGH){//if sensor doesn't read metal plate, then start counter.
        count = counter++;//this block of code counts up by '1' every second
        Serial.println(count);
        delay(1000);
   }//close conditional statement for HIGH sensor
  if (sensorStatus==LOW){//if sensor reads metal plate, reset timer to zero
        counter = 0;
   }//close conditional statement for LOW sensor
  if (counter==20){
        digitalWrite(motor1, LOW);
        digitalWrite(motor2, LOW);
        exit(0);//ends everything when timer hits max time of 77 seconds. restart needed.
   }//close termination condition
}//close void loop()
/**********************************************************************************/
```

Figure 7
Arduino Program for String Saw

Ultimately, both motors turn on once power is applied. The sensor for the string tension is read and if it’s too tight, a counter begins to count to 20. This function determines whether or not to kill power to the motors given the fact that if the string stays in one place for too long, the crystal may be damaged as a result. If the saw begins to move before the counter reaches 20, the counter resets to zero. In this example, a string saw control that includes a digital off-the-shelf controller, a number of logic components, timers, etc. can be reproduced with a few lines of code and a microcontroller.

3.3 A More Hands-On System

The string saw would also greatly benefit from a robot in order to assist in the moving of ingots as they are sliced. A pick-and-place manipulator has the capability to move in three-axis of movement due to the fact that it can move in the horizontal plane, the vertical plane, and can rotate its gripper [1]. Although, they are generally utilized in small parts assemblies, with a bulkier build and stronger motors, there would be no issues in developing a unit which could move the ingot block throughout the cutting process. Lower-level languages, such as machine or assembly language, could be utilized for this task; however, a higher-level language would also work. Higher-level languages are more "procedure-oriented" because they allow for programmers to concentrate on the issue at hand rather than the smaller details (such as checking data values, lengths, movement, etc.) [1].

Chapter 4 – Environmental Control

4.1 Current Dry Room Environmental Control

Within the Crystal Growth Department, the dry room is the most important room due to the fact that the crystals are stored, measured, cut, and installed into their frames within this room. Sodium Iodide crystals used in radiation detection devices are extremely hygroscopic. The polishing and assembly of the sodium iodide crystals into a non-hygroscopic atmospheric assembly is required to be done in an extremely dry atmosphere containing a minimal amount of water molecules. The dry room provides such an environment. The room's air is circulated first through a cooling coil to remove moisture, then through a desiccant wheel to absorb the remaining moisture, and back into the room.

Because of this, the environment control is very important. Currently, there is a simple thermometer installed. The development of an inexpensive, sophisticated wireless temperature sensor network and digital thermostat technology for the HVAC (heating, ventilation, and air conditioning) system would greatly aid in monitoring and controlling the current environment in this room.

Recent advancements in microcontrollers and form factor compatible radio modules have enabled the development of new, low-cost, programmable wireless technologies. HVAC systems have become increasingly complex and important in recent years with the growing cost of energy in the global market. The objective of this portion of the project is the creation of a simple, efficient, and cost effective means for HVAC control. The thermostat proposed for this project uses compact radio modules driven by a microcontroller to allow for temperature measurements to be taken from multiple locations simultaneously and relayed wirelessly to the control panel. Wireless temperature sensors are beneficial to HVAC system control for several reasons. Temperature readings are not restricted to collection from single location nor confined to collection from the thermostat control panel. A sensor may be placed in the most desirable location for temperature measurement regardless of the control panel placement. Multiple sensors may be placed around the large room for more precise control over temperature regulation.

4.2 Proposed Dry Room Environmental Control

Currently, the control is on ON/OFF. In this system, a sensor detects fluctuations and turns on or stays off based on these fluctuations. The proposed system would implement proportional-derivative-control (PID). A PID controller is a control loop feedback device widely used in industrial control systems. A PID controller calculates an "error" value as the difference between a measured process variable and a desired set point. The controller attempts to minimize the error in outputs by adjusting the process control inputs [1].

The PID controller algorithm involves three separate constant parameters: the proportional, the integral and derivative values, denoted P, I, and D. Simply put, these values can be interpreted in terms of time: P depends on the present error, I on the accumulation of past errors, and D is a prediction of future errors, based on current rate of change. The weighted sum of these three actions is used to adjust the process via a control element such as the position of a control valve, a damper, or the power supplied to a heating element [1].

$$\mathrm{u} = \underbrace{K_p e}_{\substack{\text{Proportional}\\ \text{Term}}} + \underbrace{K_i \int_0^t e\,dt}_{\substack{\text{Integral}\\ \text{Term}}} + \underbrace{K_d \frac{d}{dt} e}_{\substack{\text{Differential}\\ \text{Term}}}$$

Figure 8
PID Equation

As shown in Figure 8, the amount of adjustment available as well as the amount of interaction for each adjustment requires the system to be well understood in order to get an appropriate response. There are optimum points for each gain factor within each system. If there is too small of an effect, there will be poor control. With too large of an effect, the system will oscillate. In the proposed design, these three factors are replicated by potentiometers so that the system may be perfectly tuned.

Another approach to the design of the controller would be to use its analog pulse-width modulation (PWM) capabilities. PWM is a modulation technique that conforms the width of the pulse based on modulator signal information. Although this modulation technique can be used to encode information for transmission, its main use is to allow the control of the power supplied to electrical devices, especially to inertial loads such as motors [1].

PWM can be implanted into a microcontroller, such as an Arduino, with minimal effort and minimal spending. The PWM signal is a digital square wave, where the frequency is constant, but that fraction of the time the signal is on (the duty cycle) can be varied between 0 and 100%. This design uses PWM for precise control of the ventilation fan speed with minimal use of power [1]. The Arduino allows for easy implementation of PWM, while requiring minimal programming.

With the proposed design, all of these PID control as well as pulse-width-modulation could be implemented into a microcontroller and the end product could more accurately control the dry room's temperature. As noted previously, many forms of hardware would also be necessary to for control, however, the cost of materials and implementation of such a system would be far less than that of an off-the-shelf system. A proof-of-concept was developed using a LM35DZ temperature sensor, a simple computer fan, XBEE radio modules (for wireless transmission), and two Arduino microcontroller boards. The system worked better than expected and proved that implementing a similar system into the dry room would surely benefit the department. A sample of the code for both the temperature sensing circuit as well as the receiver circuit can be seen in Chapter 7.

Chapter 5 – Conclusion

5.1 Summary

The development of scintillation crystals is extremely important in the manufacturing of PET and SPECT scanners due to the fact that these machines save many lives every year by finding and diagnosing diseases. Because of this, the quality of the crystals is also important and they are held to the highest standards by Siemens as well as the FDA. To this day, many are rejected due to imperfections that occur during the manufacturing process. As proposed by this paper, computer science can play a major role in alleviating many of these issues. Revamping the process control to utilize KNN algorithms, controlling the string saw with digital logic rather than analog, and updating the environmental control of the dry room with a smarter system would surely benefit the department. By implementing these software-based automation techniques rather than the current analog systems, control of the manufacturing process would be much more efficient.

5.2 Importance of Verification

Verification is essential in many of today’s software systems. There have been many instances were devices have failed due to poor software and the circumstances were dire. An example of this is the Therac-25. This was a radiation therapy machine which was developed with minimal software support. Between June 1985 and January 1987, the Therac-25 mistakenly delivered six massive overdoses of radiation, resulting in serious injuries and even death. It was later found that the actual testing had been minimal and that virtually no meaningful analysis of the software had been performed [6]. Because the concepts presented in this paper are in the realm of healthcare, and many are checked by organizations such as the FDA, this only further justifies the need for verification. Many of the systems could be verified using automated theorem proving (or ATP) methods. An example of this could be the string saw. Currently, the saw is controlled by an analog based system. Implementing a microcontroller and using ATP methods to verify its reasoning would greatly benefit in advancing in the saws capabilities. Breaking the process down into simpler components would be the first step in analyzing this system. A broad description of the saw's operation is as follows:

- After pressing "Start" button, motor begins to spin and saw begins cutting crystal. This is state 1 to state 2.
- Saw continues cutting crystal (continues in state 2).
- Saw aborts and motor turns off on three conditions (returning to state 1): if the "Off" button is pressed, if crystal is fully cut (limit sensor is tripped), or if a timer runs out. The timer function is essential in protecting the integrity of the crystal in cases where the motor moving the saw fails and the motor turning the cutter continues. This can cause serious damage to the crystal and in many cases it would need to be discarded.

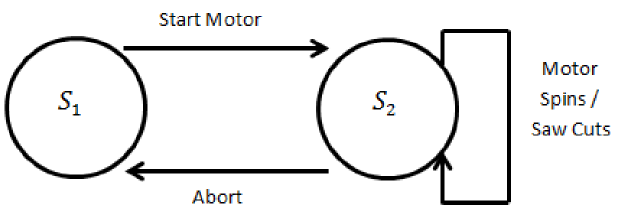
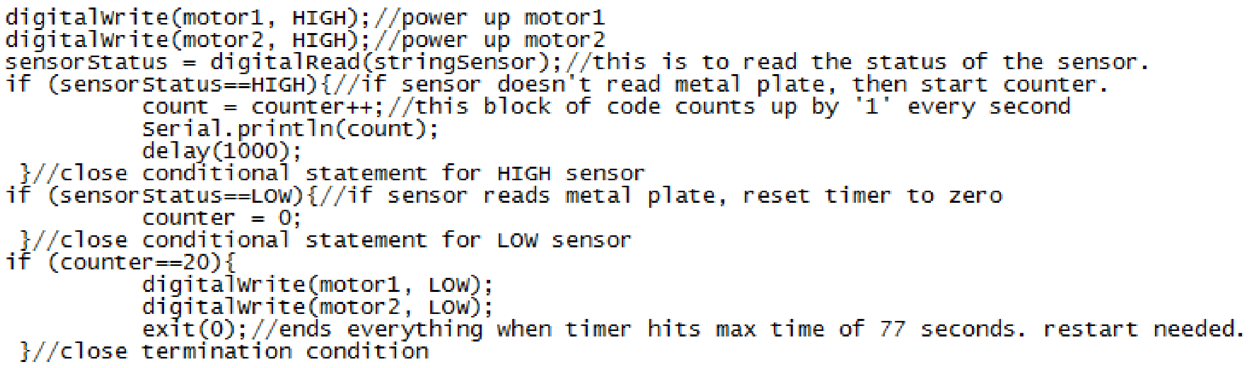


Figure 9
Kripke Structure of String Saw

Figure 9 displays a simple Kripke structure of the saw. Essentially, the system can be described with predicate logic, or first-order-logic. The system deals with HIGH/LOW's throughout; however, there are other factors, such as time in the timer, that contribute to the overall system. I believe that this would be a combination of classical logic as well as modal logic, where time is involved. An example of a program for this system can be seen in Figure 10.



```
digitalWrite(motor1, HIGH);//power up motor1
digitalWrite(motor2, HIGH);//power up motor2
sensorStatus = digitalRead(stringSensor);//this is to read the status of the sensor.
if (sensorStatus==HIGH){//if sensor doesn't read metal plate, then start counter.
          count = counter++;//this block of code counts up by '1' every second
          Serial.println(count);
          delay(1000);
 }//close conditional statement for HIGH sensor
if (sensorStatus==LOW){//if sensor reads metal plate, reset timer to zero
          counter = 0;
 }//close conditional statement for LOW sensor
if (counter==20){
          digitalWrite(motor1, LOW);
          digitalWrite(motor2, LOW);
          exit(0);//ends everything when timer hits max time of 77 seconds. restart needed.
 }//close termination condition
```

Example Code of String Saw
Figure 10

In the code, it is apparent that the system relies heavily on logic for control. When the motors power on, they continue to run until power is cut (which is beyond the control of the logic), if the limit sensor is switched, or if the timer runs out. The limit sensor (`stringSensor`) is read using the `digitalRead` command. It is then stored in `sensorStatus` and this is where the system determines the state of the sensor. With the Boolean relations of HIGH or LOW implemented into the if-statements, the program can determine which action to take. Similarly, the counter (stored in count) enables a timer to begin once the position of a metal plate (for the hall-effect sensor) is no longer being read. This portion of the system will automatically cut power to the saw if the counter runs out – in such circumstances, this is generally the cause of motor gear slipping and the saw cutting away at only one portion of the crystal. An automated theorem proof of this process would omit any errors that can result from a poorly tested system and could potentially save the crystal being sliced, which in turn could save the company thousands of dollars. It would also be more cost-efficient than checking the system by hand. In a more complicated design, possibly one implementing machine learning techniques, ATP would also be very beneficial in verifying that all operations are correctly employed in the program. Because this is essentially an embedded system, a tool such as Dafny would not be sufficient enough on its own to verify it. According to [7], FreeRTOS is an adequate tool for testing quality assurance aspects of real-time and embedded systems. Because Dafny was created in order to prove program correctness, it would be able to work closely with FreeRTOS in order to prove the latter’s modules.

5.3 Automation and Jobs

This paper has presented the benefits of automating some of the processes within the Crystals Department at Siemens Healthcare. One thing that has not yet been mentioned, however, is the danger that automation would bring to the jobs of many of the operators currently employed within the department. According to [8], 47% of workers in America have jobs at high risk of potential automation. It was also concluded that “recent developments in machine learning will put a substantial share of employment, across a wide range of occupations, at risk in the near future" [8]. This presents a clear insight into what may occur if a higher level of automation is introduced to the department. Currently, there are at 2-3 operators working 24/7 in order to monitor growth, move crystals during cutting, and many other tasks. Since these tasks are routine, however, it is certain that they can potentially be fully replaced by the concepts and ideas presented in this paper. One must certainly take this into account before moving forward.

Chapter 6 – Bibliography

[1] N. M. Schmitt, R. F. Farwell, "Understanding Automation Systems," *Texas Instruments Incorporated*, 1984, pp. 133-225.

[2] C. Stewart, "SPECT (single photon emission computed tomography) scan," [Online]. Available: http://www.mayfieldclinic.com/PE-SPECT.htm. [Accessed: May 8, 2017].

[3] MTI Corporation, "Czochralski Process," [Online]. Available: https://www.mtixtl.com/xtlflyers/Czochralski.doc. [Accessed: May 9, 2017].

[4] M. Kirk, "Thoughtful Machine Learning," *O'Reilly Media*, 2015, pp. 15-30.

[5] D. W. Patterson, "Introduction to Artificial Intelligence and Expert Systems," *Prentice-Hall*, 1990, pp. 359-365.

[6] G. J. Holzmann, "Code Craft," *IEEE Computing Edge,* May 2017, pp. 15.

[7] M. J. Matias, "Program Verification of FreeRTOS Using Microsoft Dafny," *Cleveland State University*, May 2014, pp. 1-5.

[8] The Economist, "Artificial Intelligence: The impact on jobs," [Online]. Available: http://www.economist.com/news/special-report/21700758-will-smarter-machines-cause-mass-unemployment-automation-and-anxiety. [Accessed: May 3, 2017].

Chapter 7 – Arduino Code for Dry Room Ventilation System

Temp Sensor

```
//Pin Declarations
const int tempPin=0;


void setup()
{
  Serial.begin(9600);
  //tempPin is an analog pin and implicitly set to input: no pinMode is
necessary

}
void loop()
{
  int rawTemperatureValue=analogRead(tempPin);

  //Analogread returns a value between 0 to 1023
  //Each tick is 5/1023 V = 0.00489
  // So (rawTemperatureValue*5/1023)*10 = celsius
  double voltageValue=(double) rawTemperatureValue*5/1023;
  double calculatedtemp = voltageValue*100;

  double linearizedTemperatureAddition=(voltageValue-0.750)*100;

  double fahrenheitTemp=(25+linearizedTemperatureAddition)*9/5+32;
  Serial.println((int) fahrenheitTemp);
  delay(500);
}
```

Receiver

```
#include <PID_v1.h>
// include the library code:
#include <Wire.h>
#include <Adafruit_MCP23017.h>
#include <Adafruit_RGBLCDShield.h>


//I’m just using the two I2C pins, +5v and GND.
// These #defines make it easy to set the backlight color
#define RED 0x1
#define YELLOW 0x3
#define GREEN 0x2
#define TEAL 0x6
#define BLUE 0x4
#define VIOLET 0x5
#define WHITE 0x7

#define PID_TUNING 1

Adafruit_RGBLCDShield lcd = Adafruit_RGBLCDShield();

//These are the PID variables. Set temp is the temperature we want (say,
75 degrees).
double setTemp;double input; double output;

PID pidControl(&input, &output, &setTemp, 88, 0, 0, REVERSE);
```

```
//this is for the fan PWM
const int fanPin=3;

//and this is for the temperature controller
const int knobPin=0;

//these are potentiometers used to tune the PID (all analog inputs)
const int kPin=1;
const int iPin=2;
const int dPin=3;
void setup()
{
  Serial.begin(9600);

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  //set up some PID stuff in here
  input=0;
  output=0;
  setTemp=78;
  pidControl.SetMode(AUTOMATIC);
  //pidControl.SetTunings(2,5,1); //kP, kI, kD
  pidControl.SetSampleTime(50); //milliseconds
  //set the fan pin to OUTPUT
  pinMode(fanPin, OUTPUT);
}

int lastTemperature=0; //don't update the LCD if the temp hasn't changed.
int lastSetTemp=0;


void loop() {

  if(Serial.available())
  {
    //First, read the temperature from the xbee
    //The temperature is read as a series of characters, so we need to
    //turn each digit of the temperature into one element of a char array
    //and then pass the char array to the function atoi(), which turns a
    //char array into an integer.

    char charray[10]={0};
    int counter=0;
    while(Serial.available())
    {
      charray[counter]=Serial.read();
      counter++;
      delay(10);
    }
    charray[counter]=0; //null terminate the character array so that
atoi() will work
    int temperature=atoi(charray);

    //lets get the set temp from the potentiometer (and scale it so it has
reasonable values)
    setTemp=map(analogRead(0), 0, 1023, 69, 200);

    input=temperature;
    //Now, print the temperature to the lcd screen (if it is different
from the last one)
    if((lastTemperature!=temperature || lastSetTemp != setTemp) &&
PID_TUNING==0)
    {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Real Temp:");
      lcd.setCursor(13,0);
      lcd.print(temperature);
      lcd.setCursor(0,1);
      lcd.print("Set Temp:");
      lcd.setCursor(13,1);
      lcd.print((int) setTemp);

      //set the backlight of the LCD depending on what the temperature is
      if(temperature>100) lcd.setBacklight(RED);
      else if(temperature<75) lcd.setBacklight(BLUE);
      else lcd.setBacklight(GREEN);
    }

    if(PID_TUNING) //in this mode, you turn the knobs until you get
cooling behavior you like
    {
     double newK=analogRead(kPin);
     double newD=analogRead(dPin);
     double newI=analogRead(iPin);

     pidControl.SetTunings(newK/10, newI/10, newD/10);

     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Obs:");
     lcd.setCursor(4,0);
     lcd.print(temperature);
     lcd.setCursor(8,0);
     lcd.print(" / ");
     lcd.setCursor(11,0);
     lcd.print(setTemp);

     lcd.setCursor(0,1);
     lcd.print("P");
     lcd.setCursor(1,1);
     lcd.print( (int)pidControl.GetKp());
     lcd.setCursor(6,1);
     lcd.print("I");
     lcd.setCursor(7,1);
     lcd.print( (int)(newI/10));
     lcd.setCursor(12,1);
     lcd.print("D");
     lcd.setCursor(13,1);
     lcd.print( (int)(newD/10));
     }

    pidControl.Compute();

    analogWrite(fanPin, (int) output);
    lastSetTemp=setTemp;
    lastTemperature=temperature;
  }
}
```