Summer Internship Project
Report

Topic

Real-Time Temperature Control and current monitoring with INA226 and PID Algorithms

Submitted by

SWASTIK SRIVASTAVA
MANIPAL INSTITUTE OF TECHNOLOGY

Under the guidance of

Mr. RAMA THEJA
Head of Opto Electronics

QUANTUM AI GLOBAL
4th and 5th Floor, Nishant House, 8-2-351, N/1, Rd Number 3, Green Valley, Banjara Hills, Hyderabad, Telangana 500034

Summer Internship 2024

Abstract

During the summer of 2024, I completed a 8-week internship at QUANTUM AI GLOBAL in the Electronics cell. The primary objective of this internship was to gain practical experience in getting familiar with various sensors used in the industry and working actively with microcontrollers and hence contribute to ongoing projects within the team. Throughout my internship, I worked on various tasks, including simulating a voltage converter circuit, interfacing I2C OLED screen, current sensor INA226, temperature sensor MAX6675, 7 Semi-Vikram HBridge with Arduino Uno and also worked on ATTINY85 controller for some modifications in a system.

I developed a strong understanding of the various sensors and the microcontroller used in the project. This project taught me to debug the problems that can be faced or is faced while building up a system. Got hands on experience of working with Arduino IDE. One of my primary contribution was the temperature control system using PID algorithm which was used by the company for the quantum memory box which was used to store photons in that entire sytem.

This internship provided me with valuable insights into the workings of a corporate hardware engineering environment and helped me refine my communication and teamwork skills. Overall, my time at QUANTUM AI GLOBAL Corporation was both rewarding and educational, laying a strong foundation for my future career in Electronics market.

Contents

Chapter 1

Objective

The primary objective of my internship was to apply and enhance my engineering skills by designing and implementing a temperature control system using a PID algorithm. This involved working with various hardware components and software tools to create an efficient and functional system. To achieve this main goal, I set several specific objectives:

Develop a Temperature Control System:

- Design and implement a temperature controller using a PID algorithm with the MAX6675 thermocouple sensor and an H-bridge circuit.
- Test and optimize the PID algorithm to maintain precise temperature control for various applications.

Interface INA226 and Display Data on OLED Screen:

- Implement the INA226 sensor to measure voltage and current Of a voltage regulator circuit and to display real-time data on I2C OLED screen, including voltage and current measurements.
- Utilize the ATtiny85 microcontroller to handle the system's OLED screen operations and ensure seamless communication with other components. It was primarily used to reduce the size of the microcontroller to make it usable on a commercial product.
- Develop code to accurately read sensor data and present it in a user-friendly format on the display.

Simulate Circuits Using LTSpice:

- Simulate a 1V to 100mV converter circuit using LTSpice to validate its functionality and performance before physical implementation.
- Analyze simulation results to refine circuit design and ensure optimal efficiency and accuracy.

Enhance Technical and Problem-Solving Skills:

- Improve my ability to troubleshoot and resolve hardware and software issues encountered during the development process.
- Strengthen my understanding of embedded systems, circuit design, and microcontroller programming.

Contribute to the Development Process:

- Collaborate with team members to integrate my work into broader projects and ensure alignment with project goals.

Chapter 2

Introduction

During the summer of 2024, I had the opportunity to intern at QUANTUM AI GLOBAL, a leading organization in Quantum industry, known for its innovative solutions in optoelectronics and quantum systems. This internship was an invaluable experience that allowed me to apply my academic knowledge to real-world projects, furthering my understanding of hardware design and microcontroller programming.

2.1

The primary focus of my internship was the design and implementation of a temperature control system using a PID algorithm. This project was critical in optimizing temperature regulation for various industrial applications, where precise temperature management is essential for efficiency and safety. Utilizing components such as the MAX6675 thermocouple sensor and an H-bridge circuit, I developed a system capable of maintaining accurate temperature control under different operating conditions.

2.2

In addition to the temperature controller, I worked on integrating an I2C OLED display and INA226 sensor to monitor and display voltage and current data.

2.2.1

The ATtiny85 microcontroller was used to drive the OLED screen, ensuring efficient communication between all system components and as well as making a compact packagable for the same. This required me to write and optimize code for accurate data acquisition and display, enhancing the system's usability.

2.3

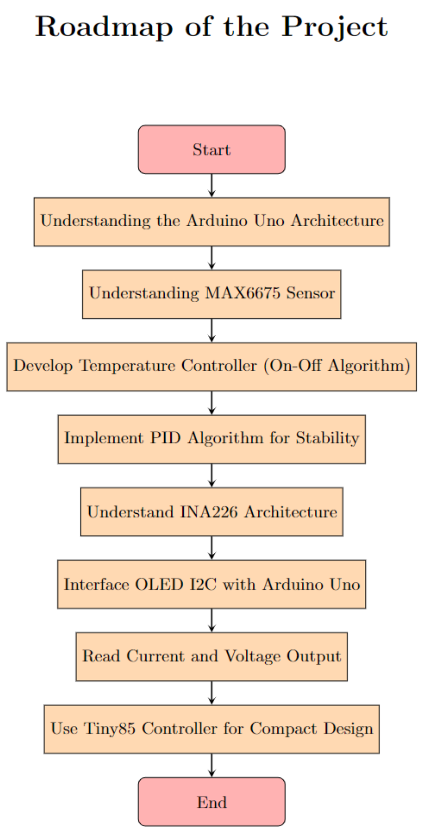
To further support the development process, I simulated a 1V to 100mV converter circuit using LTSpice, allowing for the analysis and optimization of the circuit before physical implementation. This simulation work was vital in validating the circuit design and ensuring its effectiveness.

Through this internship, I aimed to strengthen my technical skills, particularly in embedded systems and circuit design, while contributing meaningfully to the projects at QUANTUM AI GLOBAL. The experience provided me with a deeper understanding of the challenges and solutions in modern electronics, preparing me for a successful career in engineering.

Chapter 3

Work Done

3.1 FlowChart of the Path Followed



Roadmap of the Project
Start
Understanding the Arduino Uno Architecture
Understanding MAX6675 Sensor
Develop Temperature Controller (On-Off Algorithm)
Implement PID Algorithm for Stability
Understand INA226 Architecture
Interface OLED I2C with Arduino Uno
Read Current and Voltage Output
Use Tiny85 Controller for Compact Design
End

Figure 3.1: Flowchart of the Project Path

3.2 Understanding the architecture of various peripherals used

3.2.1 Arduino Uno

One of the first tasks during my internship was to gain a comprehensive understanding of the architecture of the Arduino Uno, as it played a pivotal role in my project. The Arduino Uno is a popular microcontroller board based on the ATmega328P, widely used in embedded systems due to its simplicity and versatility.

Key Components and Features

- Microcontroller (ATmega328P):

The Arduino Uno is powered by the ATmega328P microcontroller, which features a

3.3 Understanding the architecture of various peripherals used

3.3.1 Arduino Uno

One of the first tasks during my internship was to gain a comprehensive understanding of the architecture of the Arduino Uno, as it played a pivotal role in my project. The Arduino Uno is a popular microcontroller board based on the ATmega328P, widely used in embedded systems due to its simplicity and versatility.

Key Components and Features

- Microcontroller (ATmega328P):

The Arduino Uno is powered by the ATmega328P microcontroller, which features a 16 MHz clock speed, 32KB of flash memory, 2KB of SRAM, and 1KB of EEPROM.

- Digital and Analog I/O Pins:

The board includes 14 digital input/output pins, 6 of which can be used as PWM outputs, and 6 analog input pins. This flexible I/O configuration allowed me to connect various sensors and actuators, including the MAX6675 thermocouple sensor and the OLED display, enabling seamless data acquisition and output control.

- Power Supply:

The Arduino Uno can be powered via USB connection or an external power supply (7-12V). Understanding the power management was important for ensuring the reliability and stability of my projects, particularly when interfacing with external components like the INA226 and MAX6675 sensors.

- Communication Interfaces:

The board supports various communication protocols, including I2C, SPI, and UART. I used the I2C interface to communicate with the OLED display, SPI protocol for the MAX6675 sensor and also I2C protocol for INA226 sensor facilitating the integration of multiple components into a cohesive system.

- Development Environment:

The Arduino Integrated Development Environment (IDE) provides a user-friendly platform for writing, compiling, and uploading code to the board. Familiarizing myself with the IDE and its libraries allowed me to efficiently develop and troubleshoot my code. give me the code to insert the flowchart.png image below the heading flowchart of the followed

Applications in the Project

Understanding the architecture of the Arduino Uno was fundamental to the successful development of the temperature control system. It enabled me to:

- Efficiently Interface Components: I was able to connect and program the MAX6675 K-type thermocouple sensor and the H-bridge for temperature regulation, leveraging the digital I/O pins and communication interfaces.
- Optimize Code: Knowledge of the microcontroller's memory and processing capabilities allowed me to write optimized code that maximized performance.
- Troubleshoot Effectively: Familiarity with the board's architecture helped me quickly identify and resolve issues related to power management, signal processing, and communication, ensuring the reliability of the system.

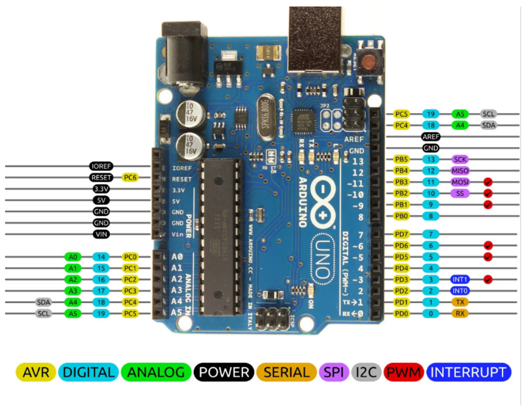


Figure 3.2: Architecture of Arduino UNO

3.3.2 INA226 Current Sensor

The INA226 is a precision current and power monitor that enables high-accuracy measurements with a digital output via an I2C interface. It is widely used in applications where power monitoring and management are crucial, such as in battery-powered devices, solar panels, and other electronic systems.

Key Features

- High-Accuracy Measurements:

The INA226 can measure both current and voltage with high precision. It calculates power by multiplying the measured current and voltage, providing real-time power consumption data.

- I2C Interface:

The device communicates via an I2C interface, allowing easy integration with microcontrollers and other digital systems. It supports standard, fast, and high-speed I2C modes, making it versatile for various applications.

- 16-Bit ADC:

The INA226 features a 16-bit analog-to-digital converter (ADC) for accurate digital representation of analog inputs. This high resolution enables precise measurements of current and voltage.

- Wide Input Voltage Range:

It supports a wide input voltage range, typically from 0 to 36 volts, making it suitable for monitoring a variety of power sources.

- Configurable Alert Pin:

The INA226 includes a configurable alert pin that can be programmed to trigger when specific conditions are met, such as overcurrent or overvoltage. This feature enhances system safety and reliability.

- Calibration:

The device allows for calibration of current, voltage, and power measurements to compensate for external component tolerances, improving overall accuracy.

Internal Architecture

- Current Sensing:

The INA226 is designed to measure current across a shunt resistor connected in series (R100 resistor i.e. 0.1 ohms in INA226) with the load. The voltage drop across the shunt is proportional to the current flowing through it, allowing the device to calculate the current.

- Voltage Measurement:

In addition to current sensing, the INA226 measures the bus voltage directly connected to the load. This dual measurement capability provides comprehensive power monitoring.

- Data Registers:

The device contains several internal registers that store measurement data and configuration settings. Key registers include the shunt voltage register (0x02), bus voltage register, current register (0x04), and power register (0x01).

- Calibration Register (0X00):

The calibration register allows users to input calibration values that scale the current and power measurements, compensating for variations in the shunt resistor and other external components.

The formulas used by the current sensor to calculate current

$$cal = \frac{0.00512}{currentLSB \times R_{shunt}}$$

$$currentLSB = \frac{MaximumExpectedCurrent}{2^{15}}$$

$$current = \frac{ShuntVoltage \times CalibrationRegister}{2048}$$

$$Power = \frac{Current \times BusVoltage}{20,000}$$

Numerical Example

- Assume 16 bits for INA226 for the ADC conversion we get $2^{15} = 65536$.

  - Consider Maximum current to be measured to be 10A.
  - Rshunt value for the INA226 = 0.1ohms.

$$Current\ LSB = \frac{10\ A}{65536} \approx 0.00015259\ A/bit$$

To find the value of the calibration register, use the following formula:

$$Calibration\ Register\ Value = \frac{0.00512}{Current\ LSB \times Shunt\ Resistor\ Value}$$

Where:

0.00512 is a fixed value from the INA226 datasheet.

Calculate the Calibration Register Value:

$$Calibration\ Register\ Value = \frac{0.00512}{0.00015259\ \times\ 0.1} \approx \frac{0.00512}{0.000015259} \approx 33554.432$$

Since the calibration register value must be an integer, round to the nearest whole number:

$$Calibration\ Register\ Value \approx 33554$$

Once the calibration register is set, calculate the current using:

$$Current(A) = Current\ LSB \times Current\ Register\ Value$$

Where $Current\ Register\ Value$ is the value read from the INA226's current register.

If the INA226 returns a current register value of 1000:

$$Current(A) = 0.00015259\ A/bit \times 1000 = 0.15259\ A$$

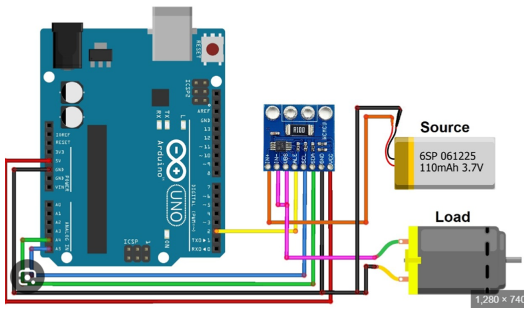


Figure 3.3: INA226 Current Sensor Architecture

Applications in the Project

- In my project, the INA226 was used to monitor voltage and current levels in a voltage regulator circuit which was used to convert a 5V input signal to a regulating 1.5-2V. The precise measurements of current and voltage allowed for:
- Real-Time Monitoring: The INA226 provided real-time feedback on the power consumption of the system, enabling dynamic adjustments and optimization of performance.
- Power Analysis: By calculating power consumption, the device allowed for detailed analysis of the system's efficiency and potential areas for improvement.
- Alert System Integration: The configurable alert pin was used to implement a safety mechanism, alerting the system in case of overcurrent conditions, which helped protect sensitive components from damage.
- Data Display: Data from the INA226 was displayed on an I2C OLED screen, giving users immediate visual feedback on system performance and aiding in diagnostics and troubleshooting.

3.3.3 MAX6675 Temperature Sensor

The MAX6675 is a highly integrated and precise thermocouple-to-digital converter designed to work with K-type thermocouples. It provides a simple and accurate solution for measuring temperature in a wide range of applications. The MAX6675 converts the analog thermocouple signal into a digital output that can be easily interfaced with microcontrollers and other digital systems.

Key Features

- K-Type Thermocouple Compatibility:

The MAX6675 is designed specifically for K-type thermocouples, which are widely used due to their broad temperature range, stability, and low cost.

- Cold-Junction Compensation:

It includes built-in cold-junction compensation to correct the voltage offset caused by the temperature difference between the thermocouple's reference junction and the measurement environment. This feature ensures accurate temperature readings over a wide temperature range.

- Digital Output (SPI Interface):

The device provides a digital output via a Serial Peripheral Interface (SPI), enabling straightforward communication with microcontrollers, FPGAs, or other digital systems.

- Temperature Resolution:

The MAX6675 offers a resolution of 0.25°C, which is suitable for most industrial and commercial applications requiring precise temperature measurements.

- Wide Temperature Range:

It can measure temperatures ranging from -20°C to +1024°C, making it versatile for various applications.

- Noise Filtering:

The device includes an integrated noise filter to reduce the impact of electrical noise on the thermocouple signal, improving the accuracy and reliability of the temperature measurement.

3.3.4 Internal Architecture

The MAX6675's architecture is designed to efficiently convert the small analog voltage from a K-type thermocouple into a precise digital temperature reading. The key components of its internal architecture include:

- Digital Processing:

Temperature Calculation: The digitized thermocouple voltage and the cold-junction compensation value are processed by an internal microcontroller to calculate the actual temperature. The result is expressed in a 12-bit digital word, representing temperature in 0.25°C increments.

- Output Interface:

SPI Communication: The MAX6675 communicates with external systems via an SPI interface. The interface consists of three primary signals: Chip Select (CS), Serial Clock (SCK), and Serial Data Output (SO). This setup allows seamless integration with microcontrollers that support SPI communication.

Data Format: The 16-bit data output from the MAX6675 is transmitted in a specific format:
Bits 15-3: Temperature data (12 bits)
Bit 2: Open-circuit fault flag
Bits 1-0: Reserved (always 0)

- A particular sensor has a clock frequency of 4.3 MHz.
- A complete serial interface protocol works in 16 cycles.

3.4 Temperature Controller without PID

In many practical applications, maintaining a specific temperature range is crucial. A temperature controller without a PID algorithm offers a simple yet effective way to achieve this and hence on-off method of control has been used to achieve this. This method is often used in systems where simplicity, cost, and ease of implementation are prioritized over the precision and adaptability offered by PID controllers.

- On-Off Control

On-off control is the simplest form of temperature control. It works by switching the heating or cooling device (which was TEC in this case) on or off when the temperature deviates from the desired setpoint.

Operation:

Heating Mode: The controller turns the TEC on when the temperature falls below the setpoint and turns it off when the temperature exceeds the setpoint-threshold point. The TEC is turned on at the setpoint-threshold temperature or less and gets turns off at the setpoint which helps TEC to get a hysteresis area and let it get stable at the particular setpoint. This whole process of turning on and off is controlled using H-Bridge.

Advantages In doing so there were some advantages which I felt doing the process. Those were

- Simple to implement and understand.
- Cost-effective for many applications.

Disadvantages But this method was not useful for long term and industrial usage.

- This lead to oscillations around the setpoint.
- Was not suitable for systems requiring precise temperature control.

3.4.1 Components of a Simple Temperature Control System

- Temperature Sensor (e.g., Thermocouple, Thermistor):

A temperature sensor is used to measure the current temperature of the system. The choice of sensor depends on the range, accuracy, and response time required for the application.

K-type Thermocouple (e.g., MAX6675): Suitable for high-temperature measurements, offering fast response times and high durability.

- Microcontroller (e.g., Arduino Uno):

The microcontroller reads the temperature data from the sensor and implements the control algorithm. It generates control signals to actuate the heater.

Analog/Digital I/O: Used to read sensor data and control actuators. Communication Interfaces: Allows interfacing with displays, communication modules, and other peripherals.

- Actuator (Heater or Cooler):

The actuator is responsible for changing the temperature of the system. This can be a heater (resistive heating element) or a cooler (fan, Peltier module).

3.4.2 Implementation Steps

- Sensor Calibration:

Calibrate the temperature sensor to ensure accurate readings. This involves comparing the sensor output with a known reference and adjusting accordingly.

- Microcontroller Programming:

Write the control logic in the microcontroller's firmware. For on-off control, implement simple logic to compare the current temperature with the setpoint and apply hysteresis. For proportional control, calculate the control signal based on the error.

- Actuator Control:

Use the control signal to actuate the heating device. This is done using H-Bridge which helps to reverse the polarity of the current.

- Testing and Optimization:

Test the system to ensure it maintains the desired temperature within acceptable limits. Adjust the hysteresis or sampling time to improve performance as needed.

3.4.3 Applications

Home Heating Systems: On-off controllers are commonly used in household thermostats for heating systems.

3.4.4 Results obtained by the above experiment

Below are the images of the results obtained by the above experiment based on the different sampling times of the MAX6675.

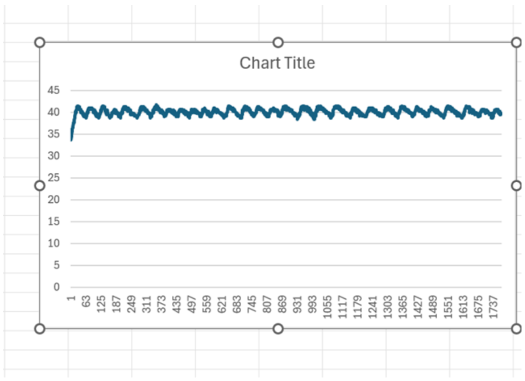


Chart Title
45
40
35
30
25
20
15
10
5
0
1
63
125
187
249
311
373
435
497
559
621
683
745
807
869
931
993
1055
1117
1179
1241
1303
1365
1427
1489
1551
1613
1675
1737

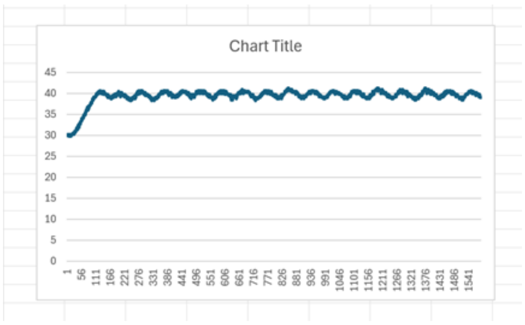


Chart Title
45
40
35
30
25
20
15
10
5
0
1
56
111
166
221
276
331
386
441
496
551
606
661
716
771
826
881
936
991
1046
1101
1156
1211
1266
1321
1376
1431
1486
1541

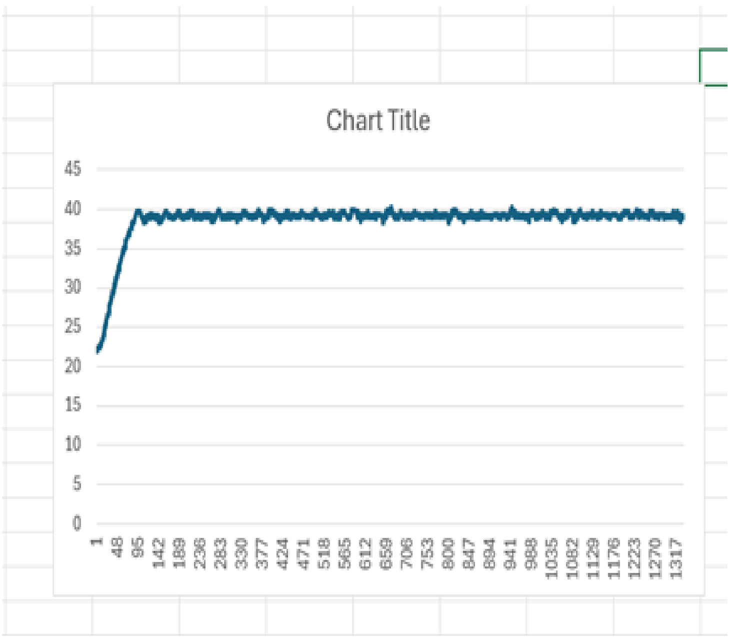


Chart Title
45
40
35
30
25
20
15
10
5
0
1
48
95
142
189
236
283
330
377
424
471
518
565
612
659
706
753
800
847
894
941
988
1035
1082
1129
1176
1223
1270
1317

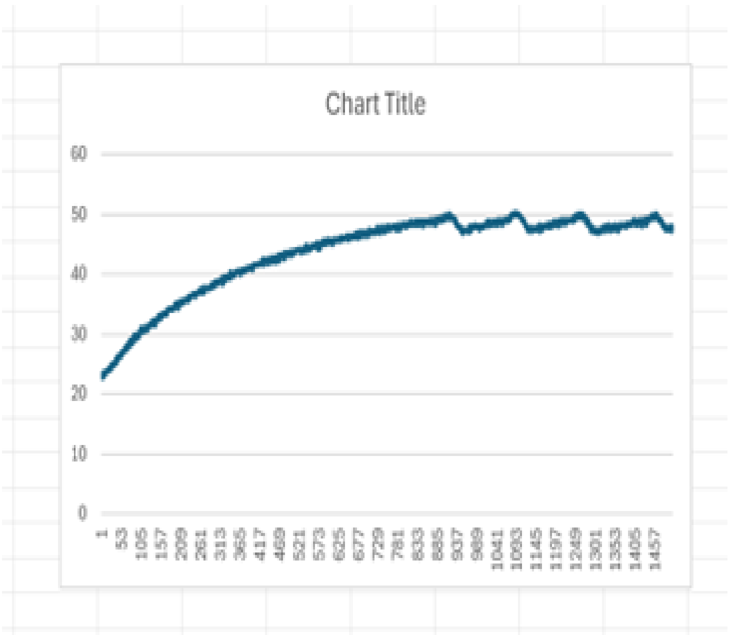


Chart Title
60
50
40
30
20
10
0
1
53
105
157
209
261
313
365
417
469
521
573
625
677
729
781
833
885
937
989
1041
1093
1145
1197
1249
1301
1353
1405
1457

3.5 Temperature Controller with PID Algorithm

In applications requiring precise and stable temperature control, a Proportional-Integral-Derivative (PID) controller provides a robust solution. The PID algorithm enhances system performance by continuously adjusting the control inputs to minimize the error between the desired setpoint and the measured temperature.

3.5.1 Theory of PID Control

The PID control algorithm is widely used in industrial control systems for its ability to provide precise control over dynamic systems. The algorithm comprises three distinct components:

- **Proportional (P) Component:**

The proportional component provides an output that is proportional to the current error value. It is calculated as the product of the proportional gain (K_p) and the error ($e(t)$), which is the difference between the desired setpoint and the measured temperature.

$$P_{out} = K_p \times e(t)$$

The proportional term helps reduce the overall error but can lead to steady-state error or oscillations if not balanced with the other components.

- **Integral (I) Component:**

The integral component accounts for the accumulated error over time. It is calculated as the product of the integral gain (K_i) and the integral of the error.

$$I_{out} = K_i \times \int e(t)\,dt$$

The integral term helps eliminate steady-state error by adjusting the control output based on the accumulated error.

- **Derivative (D) Component:**

The derivative component predicts future error by considering the rate of change of the error. It is calculated as the product of the derivative gain (K_d) and the derivative of the error.

$$D_{out} = K_d \times \frac{d}{dt}e(t)$$

The derivative term helps reduce overshoot and improve system stability by dampening the rate of error change.

Combined PID Control Equation

The overall PID control output is the sum of the proportional, integral, and derivative components:

$$u(t) = P_{out} + I_{out} + D_{out}$$

This control signal $u(t)$ is used to adjust the system's actuator to minimize the error.

3.5.2 Components of a PID Temperature Control System

The PID temperature control system consists of the following components:

- **Temperature Sensor (e.g., MAX6675):**

The temperature sensor provides accurate real-time temperature data to the microcontroller. The MAX6675 thermocouple was used to measure temperature in the project, offering high precision and reliability.

- **Microcontroller (e.g., Arduino Uno):**

The microcontroller executes the PID control algorithm, calculating the control signal based on the temperature readings and the desired setpoint.

- **Actuator (Heater or Cooler):**

The actuator, such as a heating element or Peltier module, receives the control signal from the microcontroller and adjusts the temperature accordingly.

3.5.3 Implementation Steps

The implementation of a PID controller for temperature control involves several key steps:

- **Tuning PID Parameters:**

The PID controller requires tuning of its parameters (K_p, K_i, K_d) to achieve optimal performance. This was done via trial error methods which lead to the finding oh PID parameters which were:// k_p=0.785, k_i=1.39, k_d=0,099

- **Programming the Microcontroller:**

The microcontroller is programmed to read temperature data from the sensor and compute the PID control signal. The code includes the PID algorithm, which calculates the control output based on the error, integral, and derivative components.

- **Integrating the Actuator:**

The control signal is used to drive the actuator, adjusting the system's temperature. The actuator is interfaced with the microcontroller using appropriate drivers or H-Bridge circuits to manage the current and voltage requirements. This actuator was unable to dissipate heat as fast as we want so a heat sink has to be used along with a fan which was regulated between the voltage range of 3.5-4V so as to remove heat which has trapped the heat during the rising edge of the temperature.

- **Testing and Optimization:**

The system is tested to evaluate its performance in maintaining the desired temperature setpoint. Adjustments to the PID parameters have been made as needed to minimize error, reduce overshoot, and enhance stability.

3.5.4 Applications in the Project

In my project, the PID controller was implemented to regulate the temperature using a Peltier module. The precise control offered by the PID algorithm enabled the system to maintain the desired temperature with minimal fluctuations and overshoot. The key benefits observed included:

- **Improved Accuracy:** The PID controller significantly reduced the steady-state error and improved the accuracy of the temperature control system.
- **Enhanced Stability:** The system demonstrated stable temperature regulation with reduced oscillations compared to the on-off control method.
- **Adaptability:** The PID algorithm adapted to changing environmental conditions, ensuring consistent performance across different scenarios.

3.5.5 Results obtained using PID Control

The best result obtained from implementing the PID controller in the temperature control system is illustrated in the figure below. This figure highlight the improved performance in terms of accuracy, stability, and response time.The primary characteristic of this result being 0.5 °C of deviation from the setpoint.

These results demonstrate the effectiveness of the PID algorithm in achieving precise temperature control and highlight its advantages over simpler control on-off method.

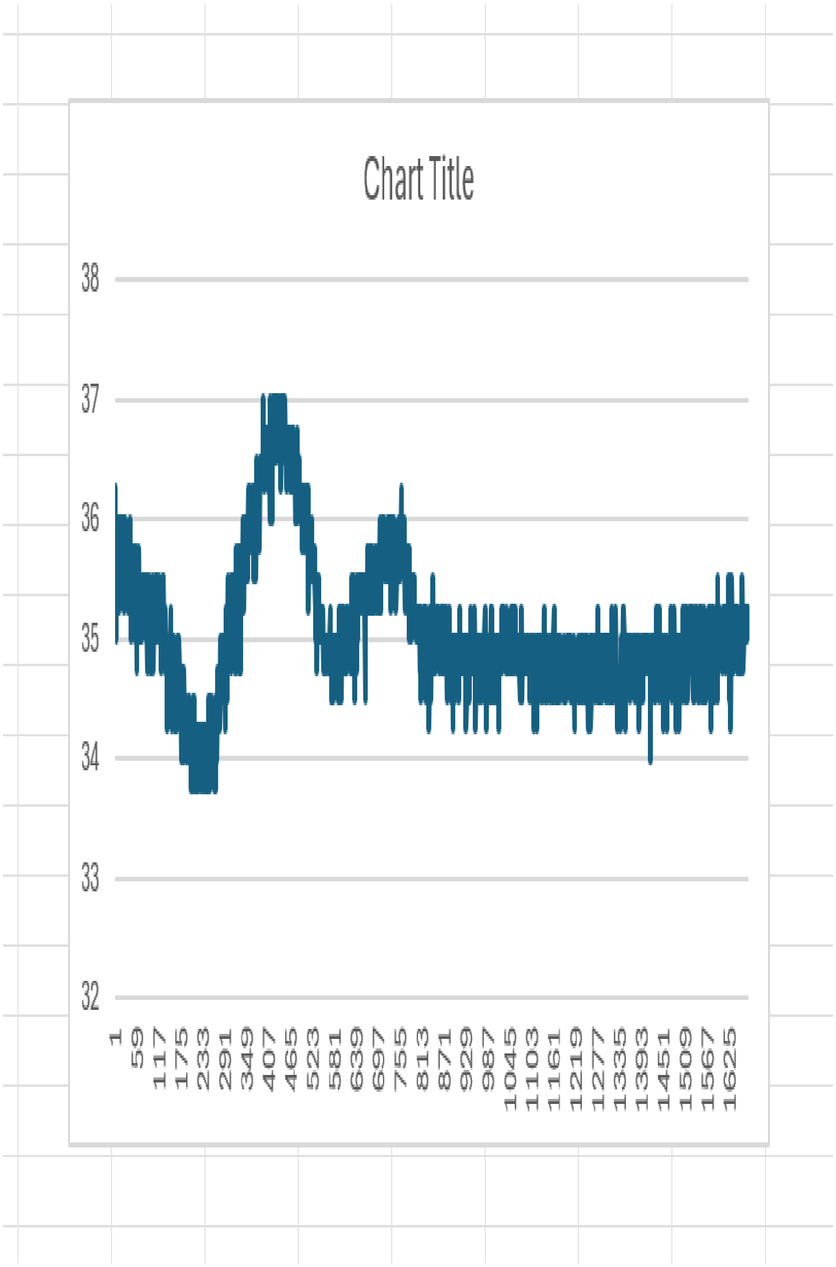


Figure 3.4: PID Best Result

3.5.6 Images of the hardware designed for the controller system

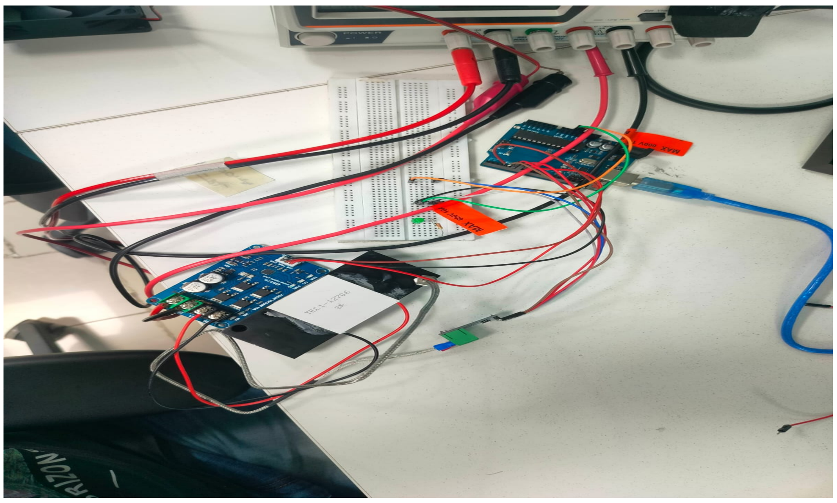
0.5 

Figure 3.5: Image for the max value

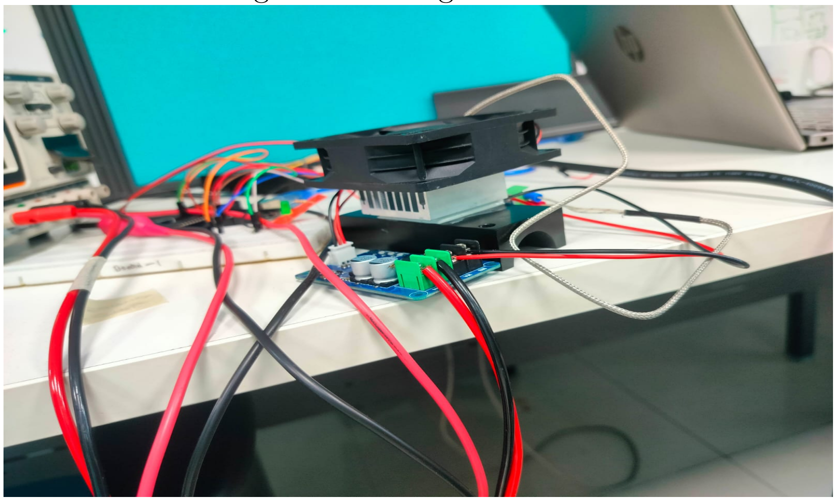
0.5 

Figure 3.6: Image for the min value

Figure 3.7: Images for the output as seen on the display

3.6 Various other side projects

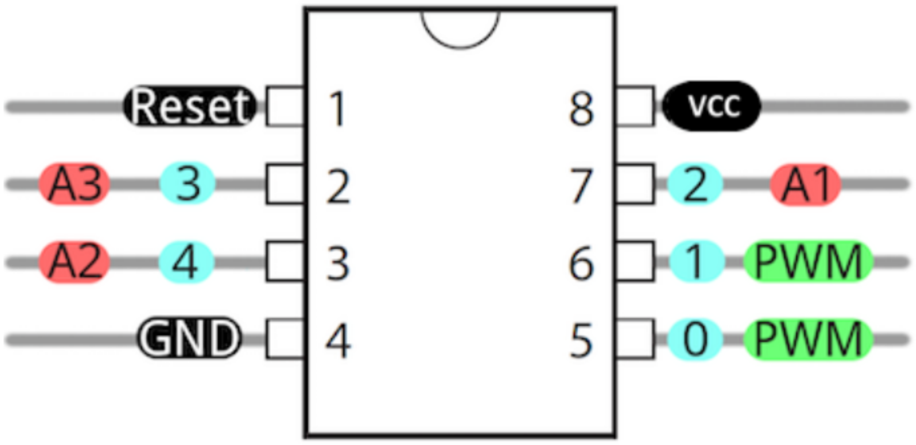
There were some side projects on which I worked on during internship. These works included interfacing INA226 sensor with I2C OLED screen to display the current and voltage values across a load and designing of a voltage regulating circuit.

3.6.1 The current sensor project

A current sensor called INA226 was used to display current and voltage on a I2C OLED screen for a 5V to voltage regulating circuit of output range 1.5-2V.
The voltage register 0X02 and the current register 0X04 was used to calibrate the required readings on the display with a resolution of upto 3 decimal places was getting displayed. The formula used to calibrate the registers of the current sensor has been defined in the architecture section of the INA226 sensor. Furhermore, in order to make the sytem compact to be able to fit on a small board I have interfaced the whole system on the ATTINY85 controller which is the smallest microcontroller existing.

ATTINY85 In order to to interface this controller with the Arduino Uno board, the following steps were followed:

- Add support for the ATtiny85 to the Arduino URL Board Manager.
- Install the ATtiny Board Package.
- Put the Arduino Uno into ISP mode.
- Connect the Arduino to the ATtiny pins.



l0.25 Pin

Diagram of ATTINY-85

- Upload the ArduinoISP program to the Arduino Uno board.
- Select the board and port.
- Choose the programmer as "Arduino as ISP" and click Burn Bootloader.

Following these steps will load the entire code in the tiny85 controller which will help to reduce the space effectively.

Some working images of the project

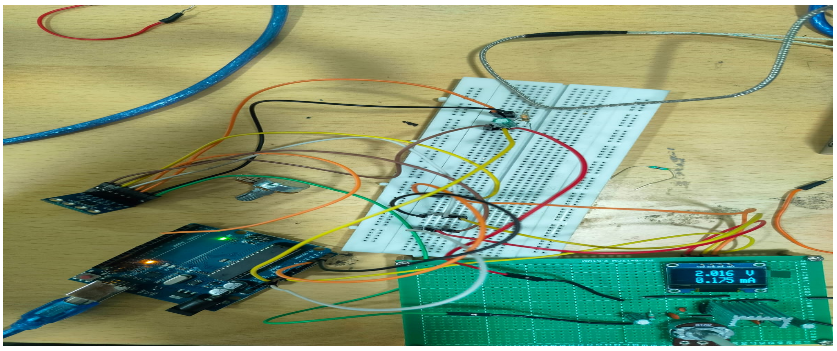
0.5 

Figure 3.8: Image for the max value

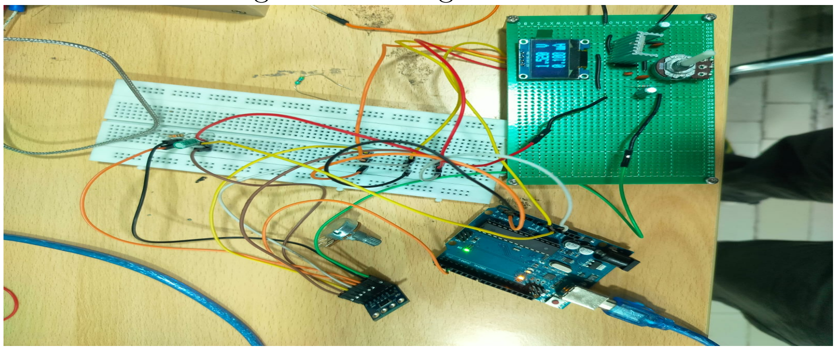
0.5 

Figure 3.9: Image for the min value

Figure 3.10: Images for the output as seen on the display

3.6.2 Voltage converter circuit

This project was based on building a voltage converter circuit from 1V-100mV value using a common emitter circuit consisting of a emmiter follower resister.

Image of the circuit used

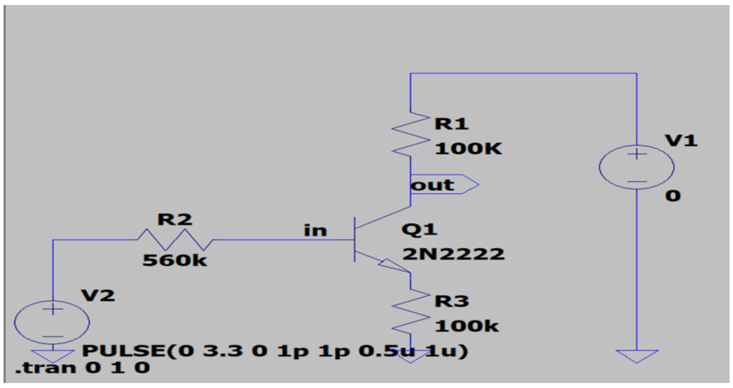


Figure 3.11: Voltage converter circuit

Results obtained through this experiment

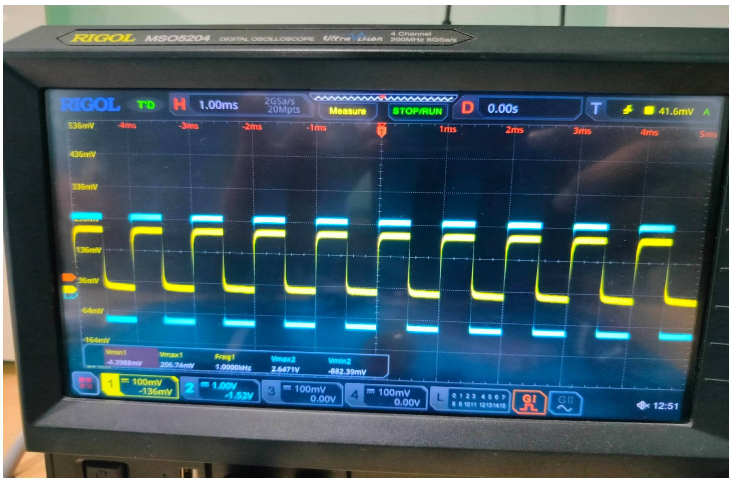


Figure 3.12: 1V-100mV results

Chapter 4

Future Work

- Now as this system was only able to work in a particular environment as the PID parameters were calculated in a particular environment so there should be a provision to autotune the whole system so that it can get stabilised on its own and the values to be set must only be used for decreasing the settling time of the system. This provision was tried to implement but it did not work out as the code was giving out some kind logical errors.This would help this product to be able to become a market based product.

- There should be a provision to automatically interface the fan and make it turn on at a particular suitable voltage on reaching the setpoint and getting turned off upon reaching a particular threshold value so that it may not be able to be operated manually and increase the heat dissipation process more efficient.

- AI Algorithms can also be used to autotune the PID parameters according to the surrounding and the set temperature which will make the system more efficient and error free and hence faster to settle down at the set temperature.

Chapter 5

Conclusion

The internship project provided valuable knowledge into the integration and application of various electronic components and systems. Through the exploration of Arduino Uno, INA226 current sensor, MAX6675 temperature sensor, significant advancements were achieved in understanding and implementing embedded systems for real-world applications.

Key Achievements

- Deep Understanding of Peripherals: Gaining an in-depth understanding of the Arduino Uno's architecture facilitated effective interfacing with components such as the MAX6675 thermocouple sensor and the INA226 current sensor. This knowledge enabled the development of a robust.

- Enhanced Data Visualization: Utilizing an OLED display to present real-time data from sensors, as well as integrating the INA226 with the OLED display for power monitoring, provided clear and actionable information. The visual feedback facilitated better system interaction and monitoring.

- Successful Implementation of Side Projects: The side projects, including the OLED display interface and temperature data logger, further expanded the application of embedded systems. These projects not only reinforced the core concepts learned but also provided additional practical experience in system design and implementation.In order to make the system compact the ATTINY85 controller was alse interfaced alongside instead the of the bigger Arduino Uno board.

Project Overview In conclusion, the development and implementation of a microcontroller-based temperature control system have demonstrated significant advancements in precision and reliability. By employing both PID (Proportional-Integral-Derivative) and on-off control methods, the system effectively regulates temperature with high accuracy, making it suitable for a variety of applications. The integration of an Arduino microcontroller with sensors such as the INA226 current sensor and the OLED I2C has enabled real-time monitoring and control, ensuring responsive and adaptive current and voltage reading.

This project explored the challenges and solutions in designing a system that maintains stability under varying environmental conditions. The use of Peltier modules as actuators provided efficient heating and cooling capabilities, which, when combined with PID algorithms, allowed for smooth and precise temperature adjustments. Through rigorous testing, the system demonstrated its ability to maintain desired temperature levels consistently, outperforming traditional control systems.

References

- https://www.renesas.com/us/en/products/microcontrollers-microprocessors/rx-32-bit-performance-efficiency-mcus/rx23e-a-thermoelectric-peltier-controller-reference-design
- http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-sample-time/
- https://www.cytron.io/tutorial/pid-for-embedded-design
- https://www.embedded.com/pid-without-a-phd/
- https://www.arxterra.com/lecture-6-pid-controllers/
- https://controlguru.com/table-of-contents/
- https://how2electronics.com/how-to-use-ina226-dc-current-sensor-with-arduino/