**Product Design**

Design and Analysis of a Robust and Portable Evaluation System for Resonant Based Silicon Photonic Biosensors [SFT]

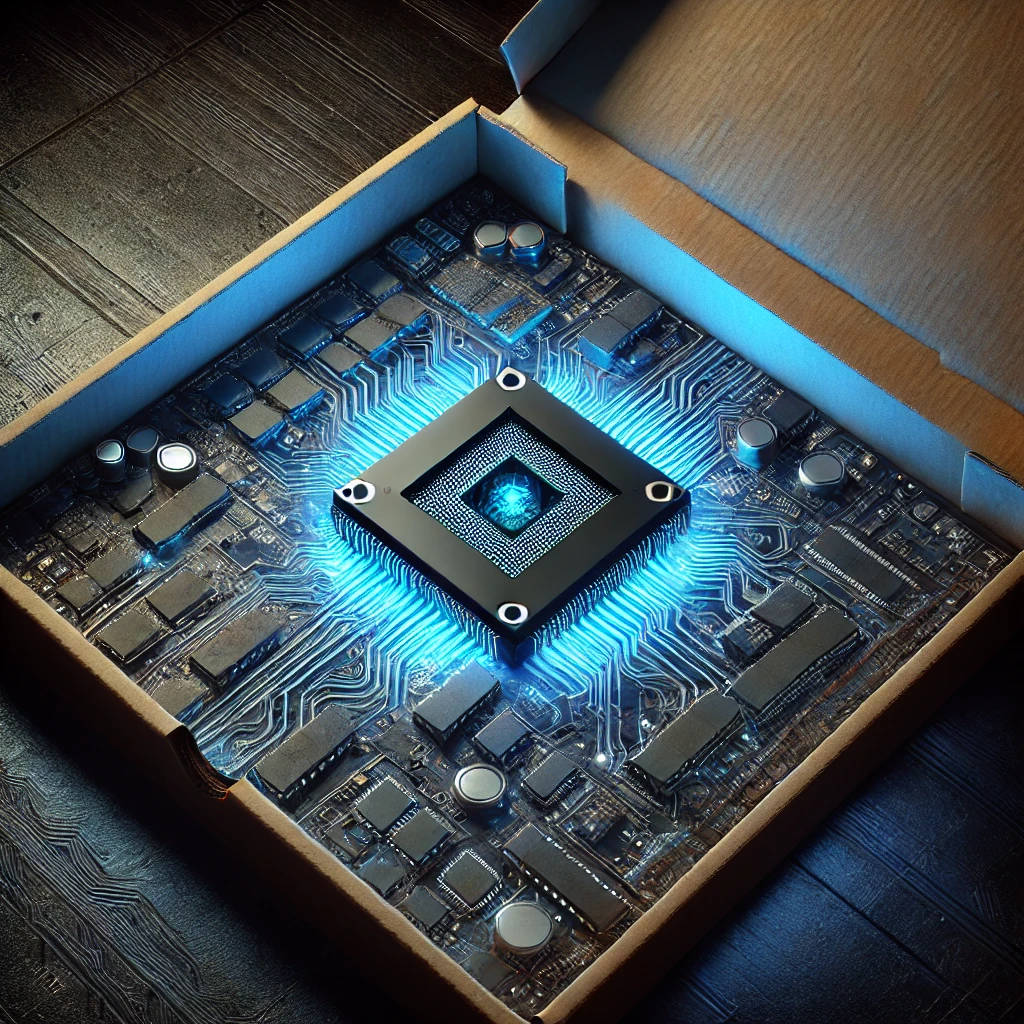
**Client:**

Faculty of Applied Science System-on-a-Chip Research lab

Ben Cohen, April 2025

**Prepared by**:

Callum O’Riley, Peter van den Doel, Tenna Yuan, Bennett Galamanga, Suhail Khalil, James Marx. [Capstone team 81]



*Figure 0.1. Imagined result chip. Source [1]*

Table of Contents

[**1 Purpose statement 3**](#_pv7gimtsayhd)

[**2 High-Level Overview 3**](#_nbe6c8j8hr74)

[2.1 Client Supplied Components 4](#_da8f7yxh5es9)

[2.1.1 Microfluidics Pump System 4](#_c0tmw3iu9z29)

[2.1.2 Microfluidic gasket 4](#_k763snvls1mt)

[2.1.3 Butterfly Laser 4](#_63gdppta1k0i)

[2.1.4 Photonic Integrated Circuit (PIC) 4](#_6uwqks6wpbdk)

[2.2 PIC Stage Sub-system 5](#_6t67yov43tdd)

[2.2.1 PIC Stage 5](#_5dt71lxegoow)

[2.2.2 Thermoelectric cooler (TEC) 5](#_qg1xlnlthuyt)

[2.2.3 Heatsink 6](#_the7zgbjr157)

[2.3 Electronic Hardware Sub-system 6](#_9ue9p0mgxcbl)

[2.3.1 Photonic Receiver Hardware 6](#_5ajd5eanhvja)

[2.3.2 PIC Temperature Controller Hardware 6](#_7vwectrxxom)

[2.3.3 Laser Driver Hardware 6](#_96rrrhhfwmd4)

[2.4 Firmware Sub-system 7](#_5g1ndrgkhxe7)

[2.4.1 Microcontroller (MCU) 7](#_l7se4tt72mc7)

[2.4.2 Firmware 7](#_2gwfl2u1cr29)

[2.5 Software Sub-system 7](#_dnfpts6zdjzz)

[2.5.1 GUI Software Interface 7](#_irtj625i05e6)

[**3 Electronic Hardware Low-level Design 8**](#_imksx61p8nw7)

[3.1 Photodiodes 8](#_89u4akuhr4q3)

[3.2 Voltage Supply 8](#_g86u7ksshas4)

[3.3 Photodiode-Transimpedance Amplifier Circuit 10](#_8cscvxuxb4po)

[3.4 ADC 13](#_yzj4me6pth0k)

[3.5 Microcontroller 13](#_fjf9cxtb28v0)

[3.6 Laser and PIC TEC Controllers 14](#_dj34atni6xle)

[3.7 Laser Driver 17](#_o5t7iw9mse3r)

[3.8 PCB Design 17](#_1lo0e67dfui)

[**4 PIC Mount Low-level Design 19**](#_6vkfo5e6ox0e)

[4.1 TEC Module 21](#_ne4082alm5t2)

[4.2 TEC Plate 21](#_gxbdcnlg5ozd)

[4.3 Aluminum PCB 22](#_7df9zfjzgxs4)

[**5 Firmware Low-level Design 24**](#_grwm98y803pu)

[5.1 Laser and Photonic Chip Interface 25](#_2a5gqmfl3c90)

[5.1.1 Laser TEC Control 26](#_njb1o29xn5ym)

[5.1.2 Stage TEC Control 31](#_ox48152r7wvi)

[5.2 Photodiode Interface 35](#_7odmqv4taf71)

[5.2.1 Configuration Mode 36](#_tv9e9slid3y1)

[5.2.2 Conversion Mode 37](#_79cnsvvb5qpu)

[5.3 PC Interface 37](#_6lk2iidxwe36)

[**6 Software Low-level Design 38**](#_ci7l1wy81nfk)

[6.1 GUI 38](#_msgyuyqr0y1z)

[6.2 Backend 42](#_z209wtpmqzxc)

[6.3 Sweeps 42](#_80pjb9299wwu)

[**7 Conclusion 43**](#_bsyuyh10rtf5)

[**8 References 44**](#_sireyc0bc86)

[**9 Appendix 46**](#_f1psfkrvdwhf)

[Appendix A: Glossary 46](#_u86xrclm6ute)

[Appendix B: PC-Firmware Interface Specification 47](#_gpahgw8nxkeq)

## 

# 1 Purpose statement

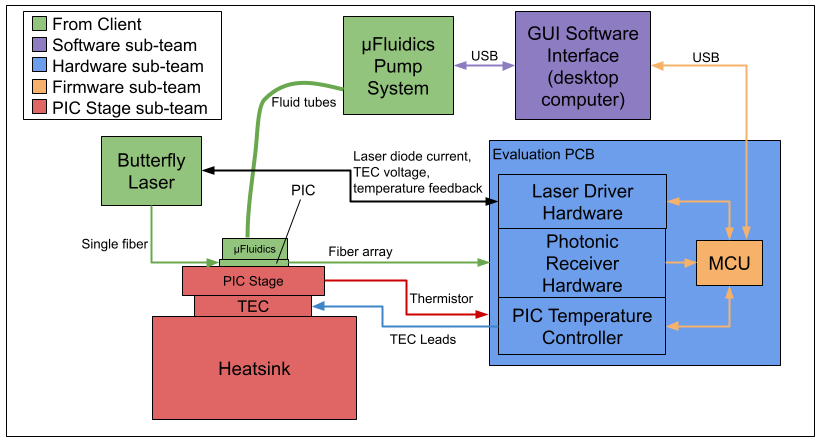
This is a capstone project to build a hardware and software evaluation system for a silicon photonic biosensor. This is a device that uses a silicon photonic chip to measure the presence of an analyte in solution delivered by a microfluidic system. The ultimate goal of the project is towards a point-of-care low-cost diagnostic system. The goal of our capstone project is building a smaller, more portable device that can evaluate different photonic chips and biological detection methods in the development of this sensing method.

# 2 High-Level Overview

The system has five main components: the software (GUI), the measurement hardware, the photonic integrated circuit (PIC), the butterfly laser and associated laser driver, and the microfluidics. Figure 2.1 shows the system architecture to integrate all these components. Our team is responsible for designing and implementing:

* The software, which the user will interface with to set sweep parameters and receive data
* The measurement hardware, which will perform the measurements of the photonic chip
* The photonic circuit mount, which will control the temperature of the photonic chip
* Firmware to communicate, control, and synchronize all the devices

This document will explain the design of each of these systems in detail.



*Figure 2.1: System Architecture*

## 2.1 Client Supplied Components

The evaluation system requires interfacing the following existing hardware and software provided by the client.

### 2.1.1 Microfluidics Pump System

This is a device to store reagents and pump them to a microfluidic gasket on top of the photonic chip through fluid tubes. It is designed and built by a graduated master’s student in our client’s lab. Our team controls it from the software system.

### 2.1.2 Microfluidic gasket

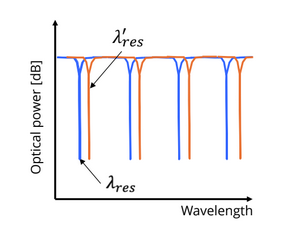
The client is providing a microfluidic gasket that moves a solution containing the analyte over the ring resonators on the photonic chip. These flows are driven by the microfluidic pump system through fluidic tubes.

### 2.1.3 Butterfly Laser

This is a low-cost butterfly laser that is controlled by the custom laser driver. The butterfly package is a standard kind of laser package that contains a built-in thermo-electric cooler (TEC) and thermistor for temperature control. Our client has identified a relationship between drive current and output light wavelength of this laser, which allows us to determine the transmission properties of the photonic chip through a range of wavelengths by sweeping through the wavelengths and measuring the output optical power.

### 2.1.4 Photonic Integrated Circuit (PIC)

This is a custom silicon photonic device created by the client. It contains ring resonators that attenuate specific wavelengths of light (seeFigure 2.2). This biosensor measures the change in resonant wavelength from depositing an analyte on the surface of these ring resonators. Frequency changes are a result of a changing index of refraction (seeFigure 2.2). This device is highly sensitive to changes in temperature, so to ensure measurement stability, we must control the temperature of it.



*Figure 2.2: Change in resonant wavelengths of ring resonator due to change in neff*

## 2.2 PIC Stage Sub-system

Photonic chips require accurate temperature control for stable readings in biosensing applications. We need to build a stage that accomplishes robust thermal control while allowing for the photonic chip to be quickly and safely swapped out for evaluation of multiple different devices.

### 2.2.1 PIC Stage

This is a mechanical part to hold the chip. It must be thermally conductive to allow heat transfer between the TEC and the chip. It also contains a thermistor so that the control system, to be discussed later, can measure and accurately control the temperature of the photonic chip. It also allows for an optical fiber array to be connected along the edge of the photonic chip for optical input/output. The stage also interfaces with the microfluidics system to flow solutions containing analytes over the photonic chip for measurement. This assembly is designed to be swappable to allow for easy evaluation of multiple photonic chips.

### 2.2.2 Thermoelectric cooler (TEC)

This is the main actuator for controlling the temperature of the photonic chip. It is driven by the control system on the evaluation board. It electrically pumps heat into/out of the chip stage.

### 2.2.3 Heatsink

This dissipates the heat from the TEC.

## 2.3 Electronic Hardware Sub-system

The evaluation system PCB (or eval PCB) is a device that measures the 16 optical power signals from the photonic chip and sends them to the computer, as well as precisely controlling the temperature of the photonic chip.

### 2.3.1 Photonic Receiver Hardware

The photonic receiver hardware receives photonic signals from the photonic chip and sends them to a connected computer. It does this by first receiving the photonic signals with photodiodes (which convert the optical input to a current output), converts the current signals to voltage signals with proportional transimpedance amplifiers, captures them with an analog-to-digital convertor, and sends them to a computer through an STM32 microcontroller. These circuits are designed to be low noise, because the analysis required for measuring the transmission of these photonic ring resonators requires precise curve fitting, the accuracy of which depends on low noise.

### 2.3.2 PIC Temperature Controller Hardware

The photonic chip temperature controller contains the hardware to control the temperature of the photonic chip, including a Wheatstone bridge, instrumentation amplifier, and ADC to measure the temperature through the thermistor. A DAC and TEC driver change the output current of the TEC that pumps heat into/out of the photonic chip.

### 2.3.3 Laser Driver Hardware

The laser driver hardware must supply a specified current to the Butterfly laser and control its temperature to accurately tune its wavelength. To do this, the hardware contains a circuit to measure the temperature of a thermistor within the Butterfly laser (Wheatstone bridge, instrumentation amplifier, and ADC), a TEC driver circuit to affect change in temperature, and a laser driver module to supply a set current to the laser.

## 2.4 Firmware Sub-system

The firmware sub-team is responsible for the microcontroller and implementing all its control and data acquisition functions for the eval PCB. They also implement the interface between the microcontroller and the computer.

### 2.4.1 Microcontroller (MCU)

This is an STM32 microcontroller that receives optical measurements through the photonic receiver hardware, and controls the temperature of the chip stage through the photonic chip temperature controller. It can connect to a host computer to communicate with the GUI Software for data logging and instrument control.

### 2.4.2 Firmware

The firmware is developed in C and commands the laser driver, communicates with the ADCs on the eval PCB and the GUI, and runs the photonic chip temperature control loop. The firmware reads the current of the photodiodes through an ADC, the temperature of the thermistor through another ADC, and runs the TEC PID control loops in an interrupt service routine.

## 2.5 Software Sub-system

The software sub-team is responsible for writing all the software required to effectively control this system. This involves coordinating wavelength sweeps from the laser, and reading data back from the photonic receiver. The software should also interface with existing lab equipment to aid in verification and validation of our custom hardware’s performance.

### 2.5.1 GUI Software Interface

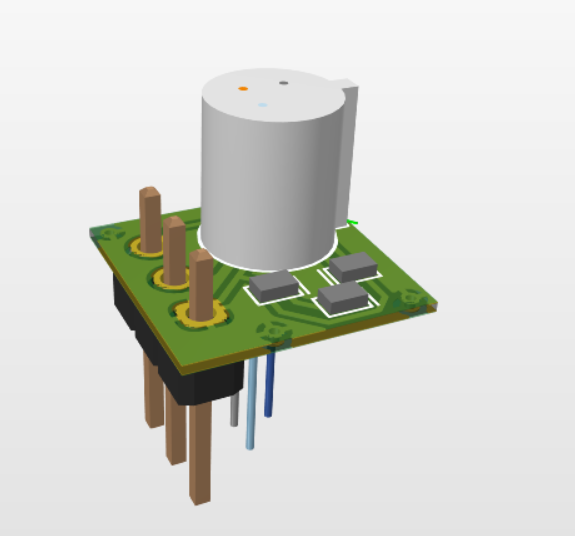
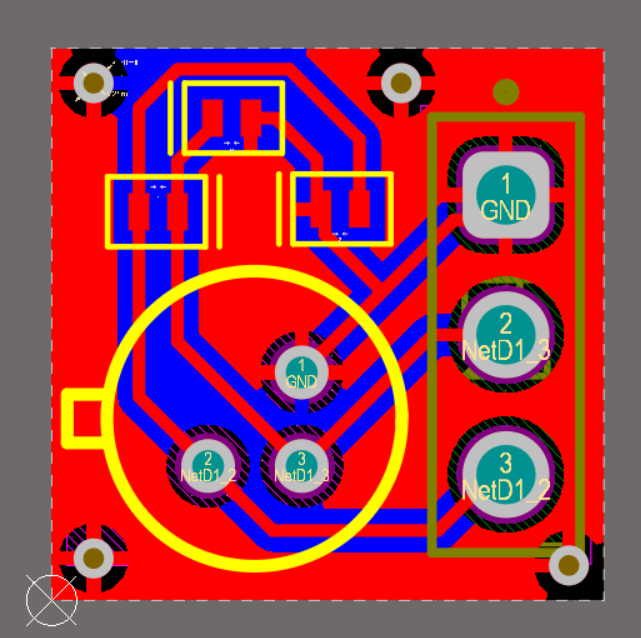
The GUI software interface runs on a lab desktop computer and coordinates the functions of all the devices in the system through a graphical user interface. For simplicity and compatibility with existing software, the GUI software is written in Python.

# 3 Electronic Hardware Low-level Design

The evaluation system PCB is a device that measures the 16 photonic signals from the photonic chip and sends them to the computer, drives a specified current through the laser, and precisely controls the temperature of the photonic chip and the laser. It also contains voltage supplies to supply power to all the onboard systems and to create precision reference voltages for the photonic receiver hardware.

## 3.1 Photodiodes

The photodiodes are set to be a specific model provided by the client (CS-1). Because they are highly sensitive to ESD, a hot-swappable adapter PCB is designed so that they can be removed and replaced quickly and without soldering if one is destroyed. The adapter PCB also contains bidirectional TVS diodes that limit the voltage across the photodiode and its package, to decrease the potential for ESD to destroy the photodiode. Figure 4.1 shows the Altium PCB design for the photodiode adapter PCB.

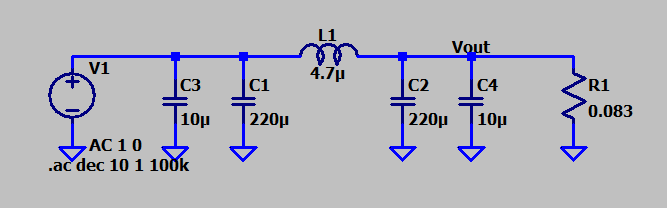


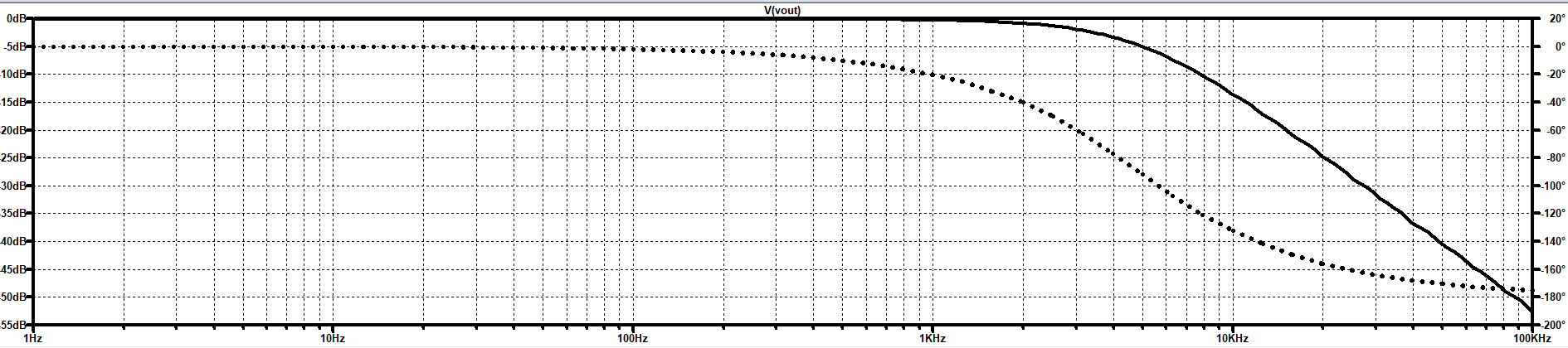
*Figure 3.1: Photodiode adapter PCB*

## 3.2 Voltage Supply

The voltage supply generates all the voltages required for the eval PCB. It uses a combination of voltage regulators and DC-DC converters to accomplish this. The voltage supply must provide 12V, 5V, and 1.8V rails to power devices on the eval PCB, and 5V and 10V precision references to bias the photodiode. The 3.3V voltage line is supplied by the STM32 Nucleo board.

The circuit is required to supply power to the TEC control circuit and the low-voltage devices, which will all draw current from a 5V rail. The TEC controller draws 1.5A, and the low-voltage devices will all likely draw no more than 500mA. A DC-DC buck converter that can supply 5V at 2A from 12V is required and the TSR 2-2450 is chosen. The input power jack is chosen to be a PJ-102A barrel jack, which is a common barrel jack used by many power supplies. An input power filter that removes any switching noise from an external 12V power supply is designed. Given that the 5V rail draws 10W and some other circuits may draw a little bit more, it is assumed that the whole circuit will draw about 1A, or 12W. A characteristic impedance of is set for the input filter to match the impedance of the circuit and a cutoff frequency of at most 10kHz (to eliminate switching noise from the power supply), and calculate the values of a Pi filter required to meet these requirements. Standard component values are chosen and some additional ceramic capacitors are added to improve filter performance at high frequencies. An LTSpice simulation of the final input Pi filter is performed as shown in Figure 4.2, showing the actual roll-off being around 4kHz, which is acceptable, as it will still attenuate 10kHz noise.





*Figure 3.2: Input Pi filter and frequency domain simulation*

On the 5V rail, capacitor filters are included to remove any output ripple from the DC-DC converter, and the photonic chip TEC controller is supplied through another Pi filter circuit given by the client to decouple the current noise from the TEC driver from the main 5V line. The LDC TEC controller is supplied through another DC-DC buck convertor passed through a filter to handle its current requirements.

A 1.8V rail is required to supply the logic interface for the photonic receiving ADC. The 1.8V regulator chosen is the MCP1700T-1802 due to its simplicity of application and the team’s familiarity with it. To get 3.3V for the STM32 logic interface, we use the onboard STM32 regulator on the Nucleo board and just supply it with 5V.

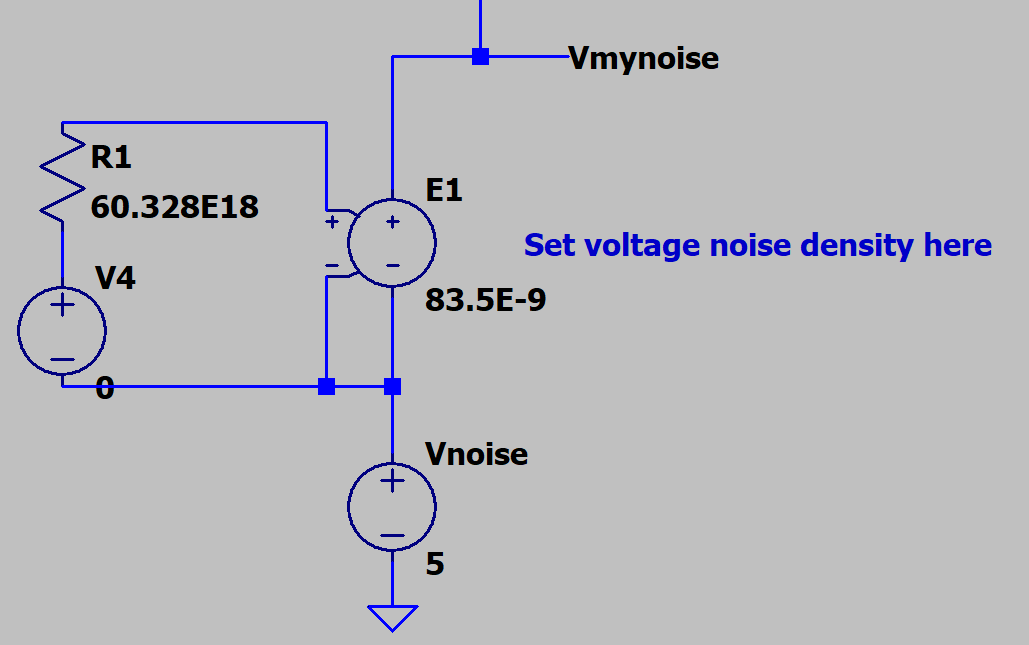
The photonic receiving side of the circuit and the LDC side of the circuit are both independently switched on/off through SPST switches on the board.

## 3.3 Photodiode-Transimpedance Amplifier Circuit

The Photodiode-Transimpedance Amplifier (PD-TIA) circuit is a circuit to convert the optical outputs of the photonic chip into voltage values that an ADC can read to achieve photonic power measurement (RS-1). BPD-70P-10B-FA photodiodes are used, per a design constraint to use existing photodiodes (CS-1). The design of this circuit is done using Analog Devices’ Photodiode Circuit Design Wizard.

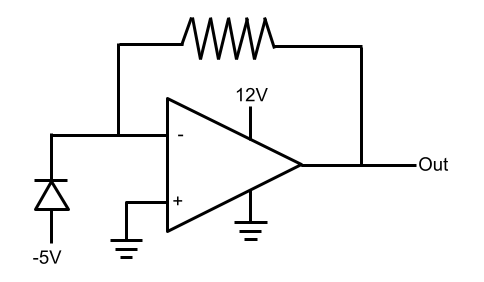
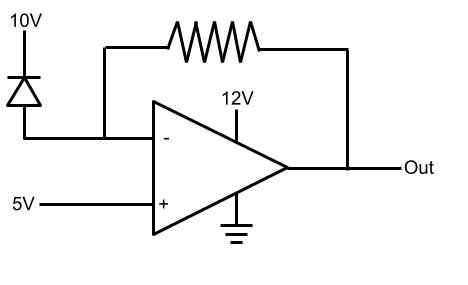
To achieve the goals of high-resolution peak detection (GS-2) and accurate peak detection (GS-3), low noise AD8657ARMZ-R7 op-amps are used in the TIAs. They are also favoured because of their low cost (CS-2). This part gives an estimated noise of 25.4μVrms and an effective number of bits of 15.6. These values inform the choice for ADC, as the ADC is desired to have a similar noise figure and ENOB. The preliminary analysis done by the design tool does not consider the effect of precision voltage reference noise, so those must be added in. The tool provides SPICE noise models that can be changed after the fact.

Noise from the precision voltage references is modelled as white noise with noise density given by component datasheets. LTSpice does not provide voltage noise sources, so one is modelled by using a resistor and a voltage-dependent voltage source (VDVS). LTSpice allows for noise modelling of resistors because the software accounts for Johnson noise. A resistor value of will give white noise density [2], so the gain of the VDVS is set to the noise density of the precision reference we are evaluating. Noise can be added on top of a regular voltage source to get the precision reference model with noise as shown in Figure 4.3.



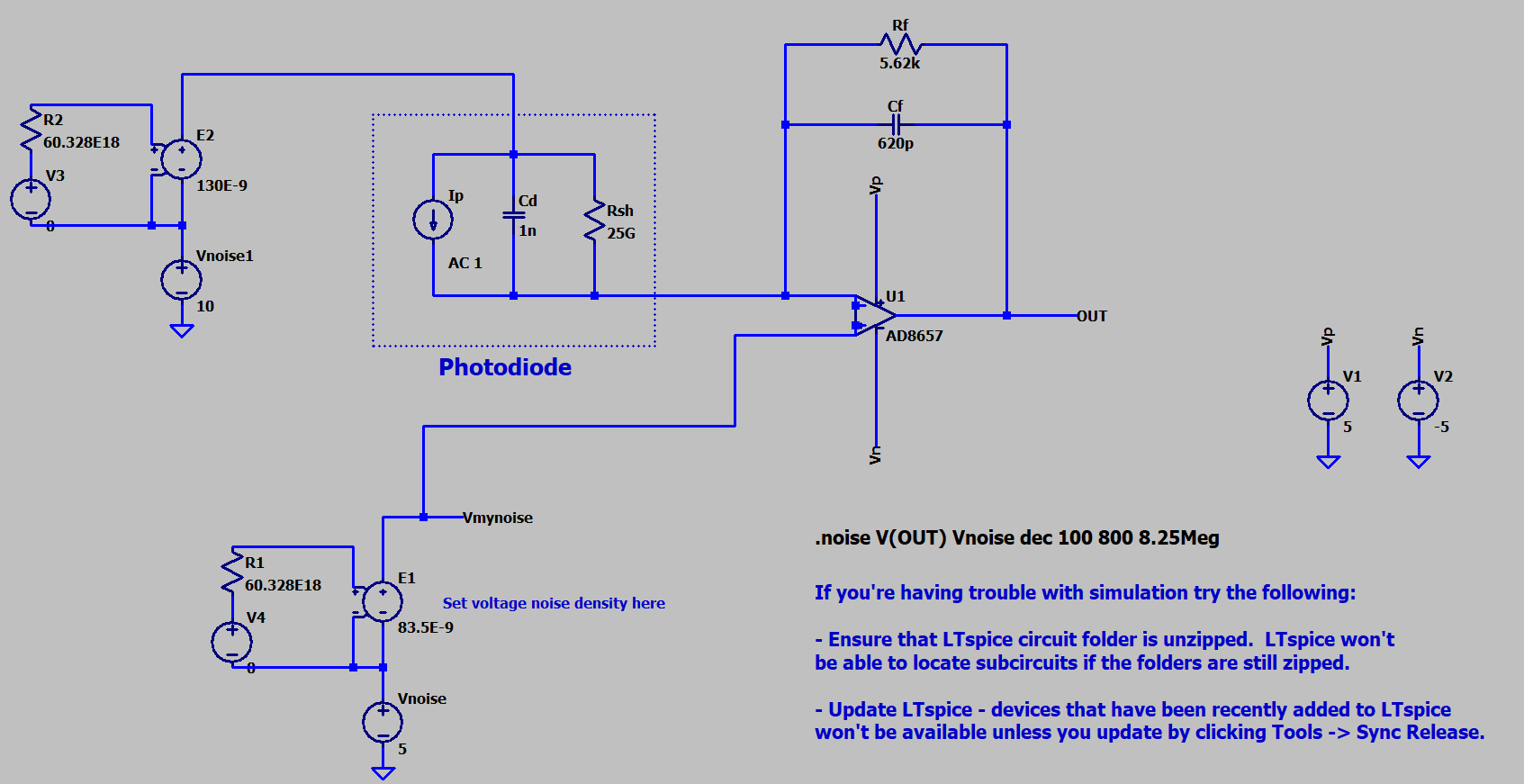
*Figure 3.3: Precision voltage reference noise model in LTSpice with voltage of 5V and noise density of*

The chosen ADC can only read signals between 0-5V. Figure 4.4 shows two options for biasing a photodiode to 5V while keeping a positive output swing for the ADC.



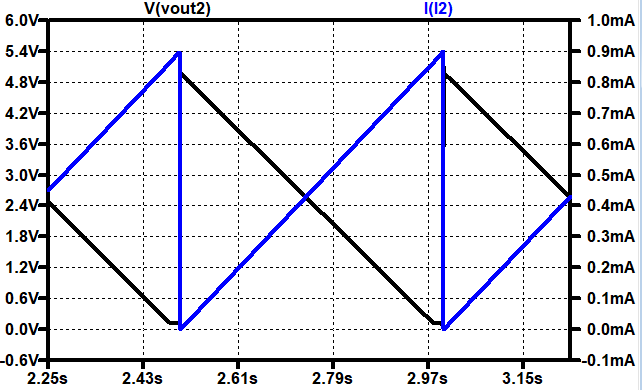
*Figure 3.4: Precision voltage reference noise model in LTSpice*

Both of these circuits are simulated in LTSpice using the SPICE models generated from the design tool with realistic values for noise for their required reference voltages as shown in Figure 4.5.



*Figure 3.5: LTSpice noise simulation*

It is found that the left circuit in Figure 4.4 yields less output noise than the right circuit. This is because typically negative voltage generators (like charge pumps) have a large amount of ripple on the output, which is coupled into the circuit through the photodiode. Therefore, development on the left circuit is continued. The output voltage and input current of this circuit are plotted together in Figure 4.6.



*Figure 3.6: Output voltage (V(Vout2)) and input current (I(I2)) plot*

The amplifier is inverting, but the output signal is shifted up to an offset of 5V to create a strictly positive output voltage that can be read by an ADC.

Using the noise analysis simulation, the LT1236BCS8-10 is chosen for the 10V reference and the ADR4550ARZ-R7 for the 5V reference, as they provide low noise at a low cost. It is found that the final RMS noise for this circuit is 389.23 µVrms.

Instead of implementing fixed gain resistors for the TIAs (5.62k, as suggested by the design tool), 100k potentiometers are used giving flexibility to increase the gain. This is beneficial if the input photonic signal level is lower than expected due to insertion loss. PS10KV50-104A3030 potentiometers are chosen for their low cost and large size, making adjustment easier.

## 3.4 ADC

The AD4695BCPZ is chosen as the main ADC for the photonic receiver, as it has a high signal-to-noise ratio (up to 93 dB) among ADCs with similar specs, and enough channels for the 16 photodiodes. The TXS0108EQPWRQ1 level shifter is used to match the logic voltages between the ADC and the MCU. It is selected for having 8 bidirectional channels, as 7 lines are required to communicate between the ADC and the MCU.

## 3.5 Microcontroller

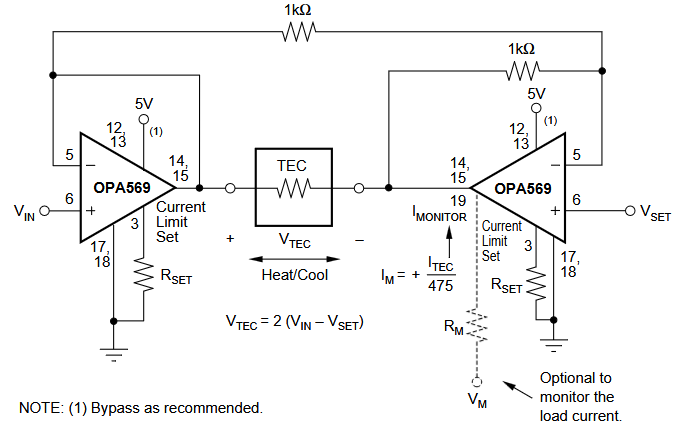
The MCU used is the STM32H723, which provides the required SPI and UART integrated peripherals to interface with external ADCs, the laser diode controller, and host PC control software. The MCU is integrated with the eval PCB through a NUCLEO board to make debugging and interfacing with client’s laser driver hardware easier. The STM32H7 series is chosen for its large memory size of 1 MB and fast clock speed of 550MHz, giving it ample bandwidth to handle real-time temperature control and data acquisition. The ability to run sweeps as fast as possible is desired, and data transmissions during those sweeps slows them down. It is calculated that for a fully autonomous sweep where we store 4096 data points (the number of current steps in the laser driver), at least 147 kB of memory would be needed, and having much more memory gives ample flexibility. This data is sent in batches at the end of each sweep.

## 3.6 Laser and PIC TEC Controllers

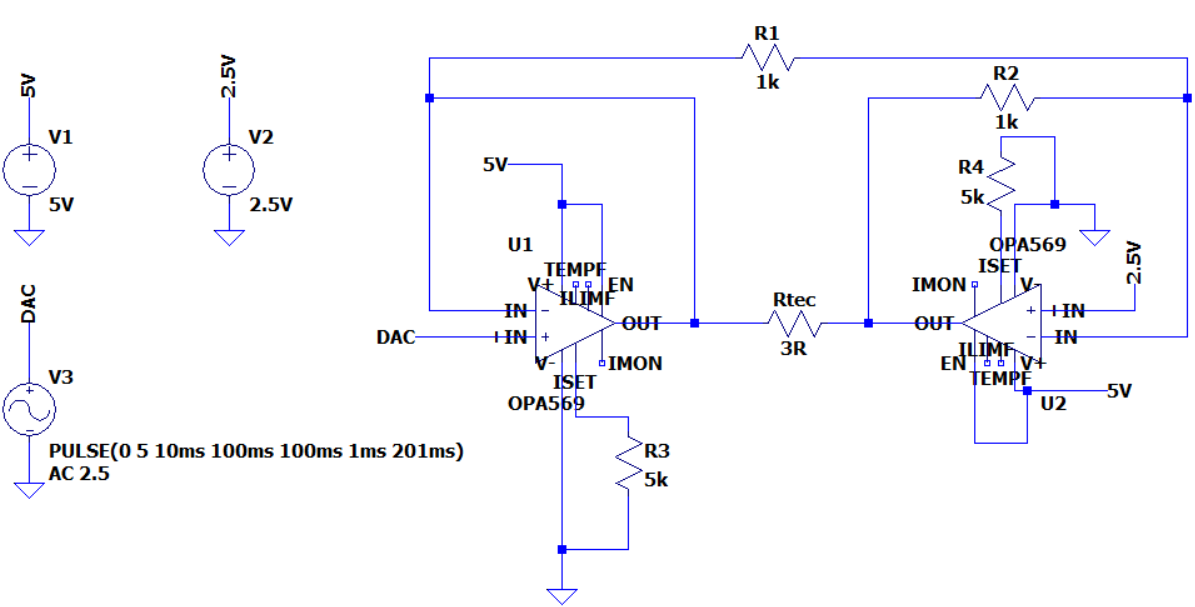
The TEC controller circuits are responsible for measuring and controlling the temperature of the chip stage (RS-4). This requires us to measure the temperature of the photonic chip or LDC, compute control action to apply using a PID controller, and apply that control action by driving current through the TEC.

The temperature of the TEC is measured using a thermistor connected to a Wheatstone bridge, INA828 instrumentation amplifier, and MCP3561 Sigma-Delta ADC. The ADC is chosen because of its high resolution (24 bits) and high sampling rate (up to 153.6 ksps). High resolution is important to be able to have a large temperature measurement range without compromising on temperature measurement resolution.

Current is driven through the TEC using OPA569 power op-amps. These are chosen for their high output current capability, relative simplicity over other TEC driver solutions, and their current limiting features. They are preferred over switching drivers because of their pure analog output, which eliminates high-frequency switching noise that could interfere with sensitive analog measurements or temperature regulation. Their inefficiency is offset by the fact that at steady-state, the control action that these parts will have to apply will be very small. The circuit topology chosen is taken from the OPA569 datasheet, and is ideal because the dual amplifier nature allows for the full supply voltage swing to be placed across the TEC module. The input voltage is set with a DAC7513 12-bit DAC, and the fixed setpoint voltage is set to be 2.5V. We can also set a current limit with a current limit set resistor, which we set to be about 2.3A (near the maximum drive capability of the op-amps but within the safe limits for the TECs) with a 5k resistor using the equation . The chosen topology is shown in Figure 3.7.

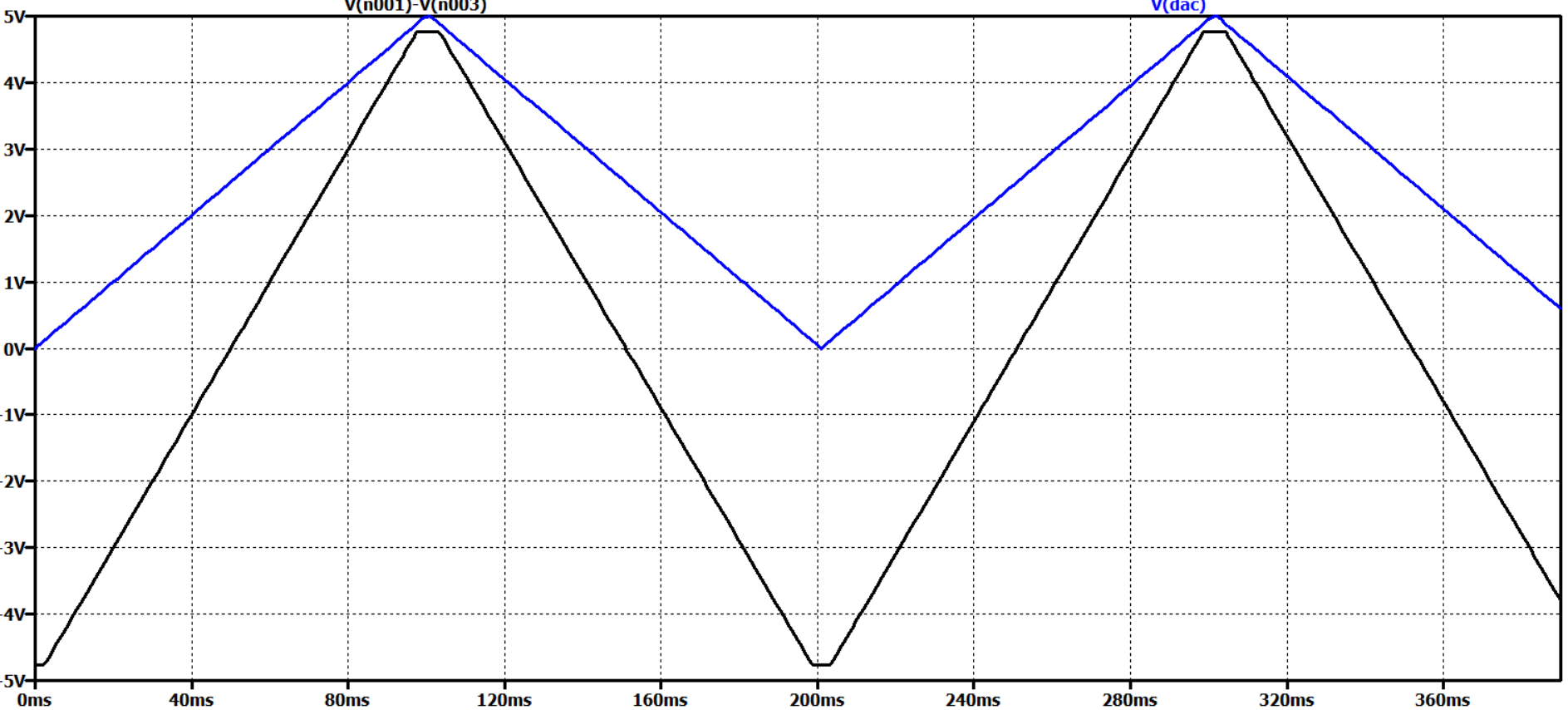


*Figure 3.7: TEC driver schematic from datasheet [3]*



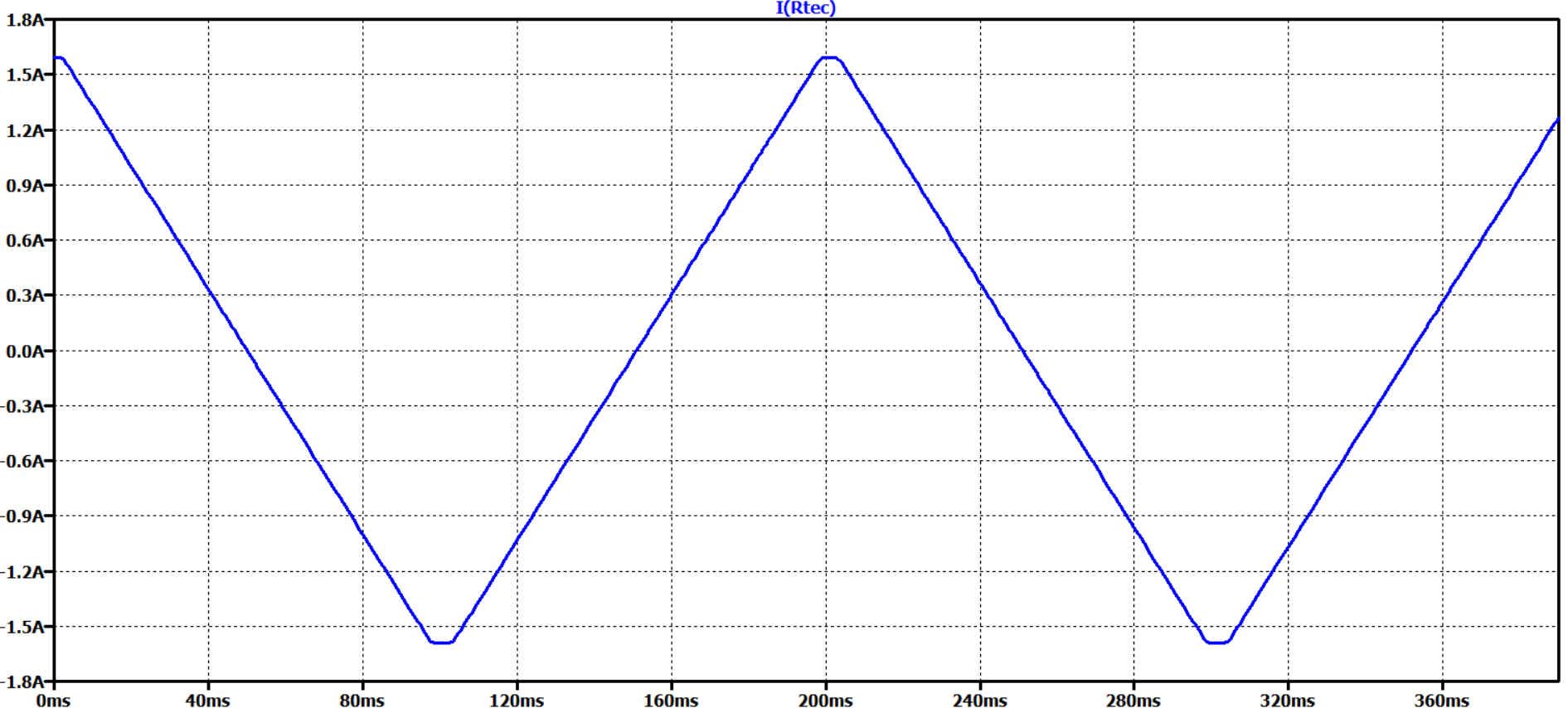
*Figure 3.8: Simulation layout in LTspice*

Figure 3.9 below shows the differential voltage across the TEC. The schematic mirrors the topology recommended in the OPA569 datasheet. Two OPA569 op-amps (U1 and U2) are used in a full H-bridge configuration to drive a 3Ω resistive load that emulates the TEC being used. A pulse voltage source (V3) models the DAC output and swings from 0 V to 5 V in a triangular waveform centered around a fixed setpoint voltage of 2.5 V (V2). This input difference directly controls the polarity and magnitude of the voltage across the TEC with . At , . When , .



*Figure 3.9: Voltage between the TEC resistor*

The figure below shows current flowing through the TEC. The current mirrors the TEC voltage waveform but scaled by the TEC resistance, which is 3 ohms, so . The plot ranges roughly between ±1.67 A, matching ±5 V across a 3Ω load.



*Figure 3.10: Current flowing through the TEC resistor*

The simulation is carried out as a transient response lasting 400 ms, starting at 10 ms with a timestep of 1 ms. The goal is to evaluate the voltage and current response of the TEC when driven by dual OPA569 power op-amps under a dynamically changing DAC control signal. This shows that the dual OPA569-based H-bridge TEC driver can smoothly regulate bidirectional current through the TEC with high linearity and full supply swing, and that it responds accurately to our full range of DAC inputs.

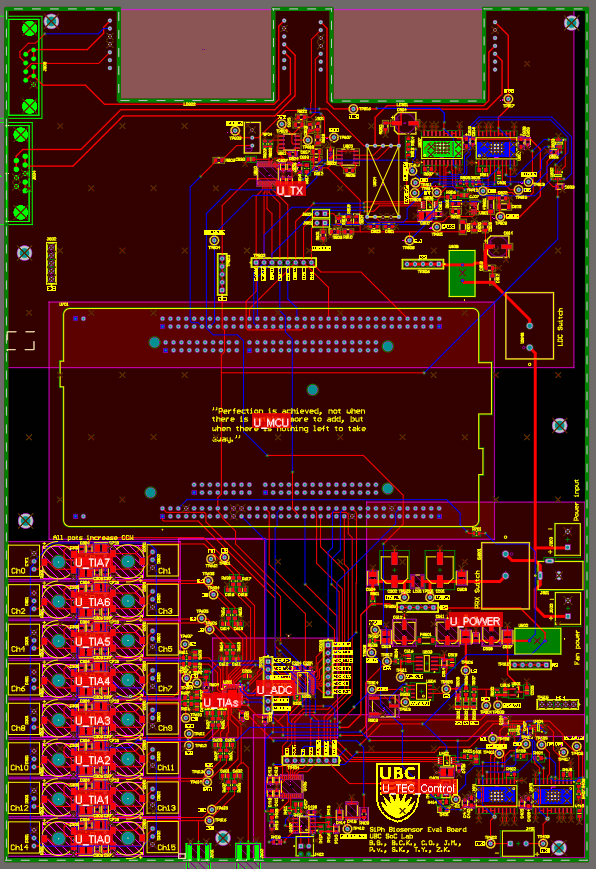
The temperature control hardware is duplicated on the board for the chip stage TEC controller and the laser TEC controller.

## 3.7 Laser Driver

We need a wavelength-tunable laser to change the wavelength of light that we apply to our device (RS-10). Changes in the butterfly laser’s input current lead to changes in the output wavelength. To accomplish this, we use a ThorLabs MLD203CLN laser diode current driver to drive a specified current (set by the STM32’s internal DAC) through the laser diode. It can be enabled/disabled by a TMUX1119DBVR electronic switch IC. This laser diode driver is chosen primarily for its low current noise, which corresponds to low wavelength noise (due to the current-wavelength tuning effect), which satisfies RS-2, RS-3, and GS-2.

## 3.8 PCB Design

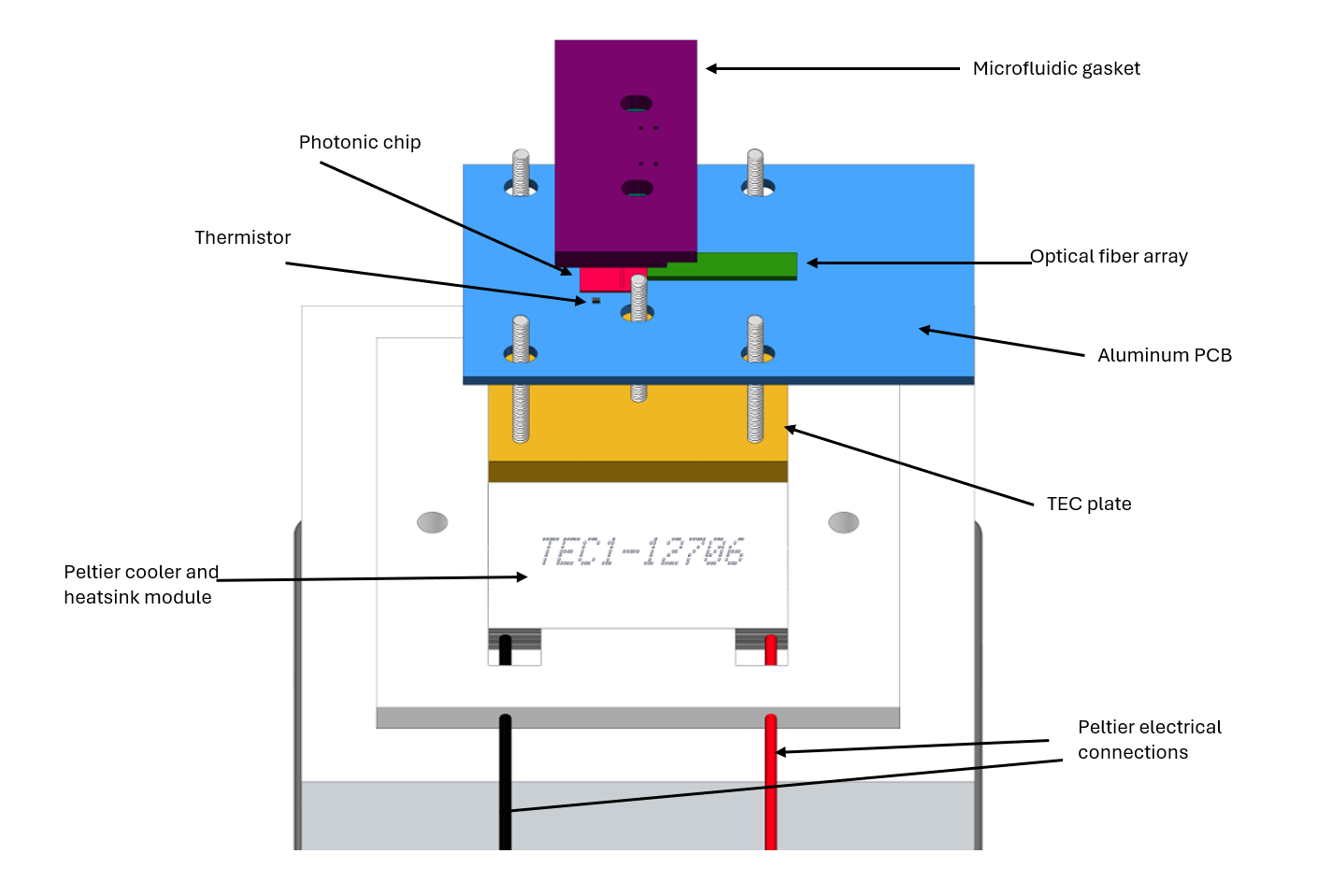
All electronic hardware is integrated onto a single PCB, designed using Altium Designer. To ensure adequate grounding with good routing freedom, a 4-layer PCB with two internal ground planes and ground stitching vias is used. Multichannel design strategies are used to easily replicate routing of the similar circuits (particularly the PD-TIA circuits). Rooms are used to group the subcomponents of the PCB design together, and the rooms are spread out across the board to prevent noisy circuits (TEC driver, SPI lines) from interacting with sensitive analog circuits. Signal integrity is important for meeting the requirements of sweep resolution (RS-2) and repeatability (RS-3). The board is divided into the photonic receiver/photonic chip temperature control hardware on the bottom and the laser diode controller on the top, straddled by the STM32 Nucleo in the middle. The PCB includes nylon standoffs to support it and slots for the butterfly laser to slot in (one where the laser is driven/controlled using our driver and one where it can be driven/controlled with a benchtop driver). The PCB design is shown in Figure 4.7.



*Figure 3.11: PCB Design in Altium*

# 4 PIC Mount Low-level Design

One of the project requirements (RS-4) is temperature control for stable readings in biosensing applications. Another requirement (RS-8) is the ability to hot swap chips quickly. The project calls to build a stage that accomplishes robust thermal control, while allowing for the photonic chip to be quickly and safely swapped out for evaluation of multiple different chips. A thermoelectric cooler (TEC) is used to pump heat into and out of the chip for temperature control, and a thermistor is used to measure the temperature. Figure 5.1 shows a labelled system diagram of the PIC Mount.

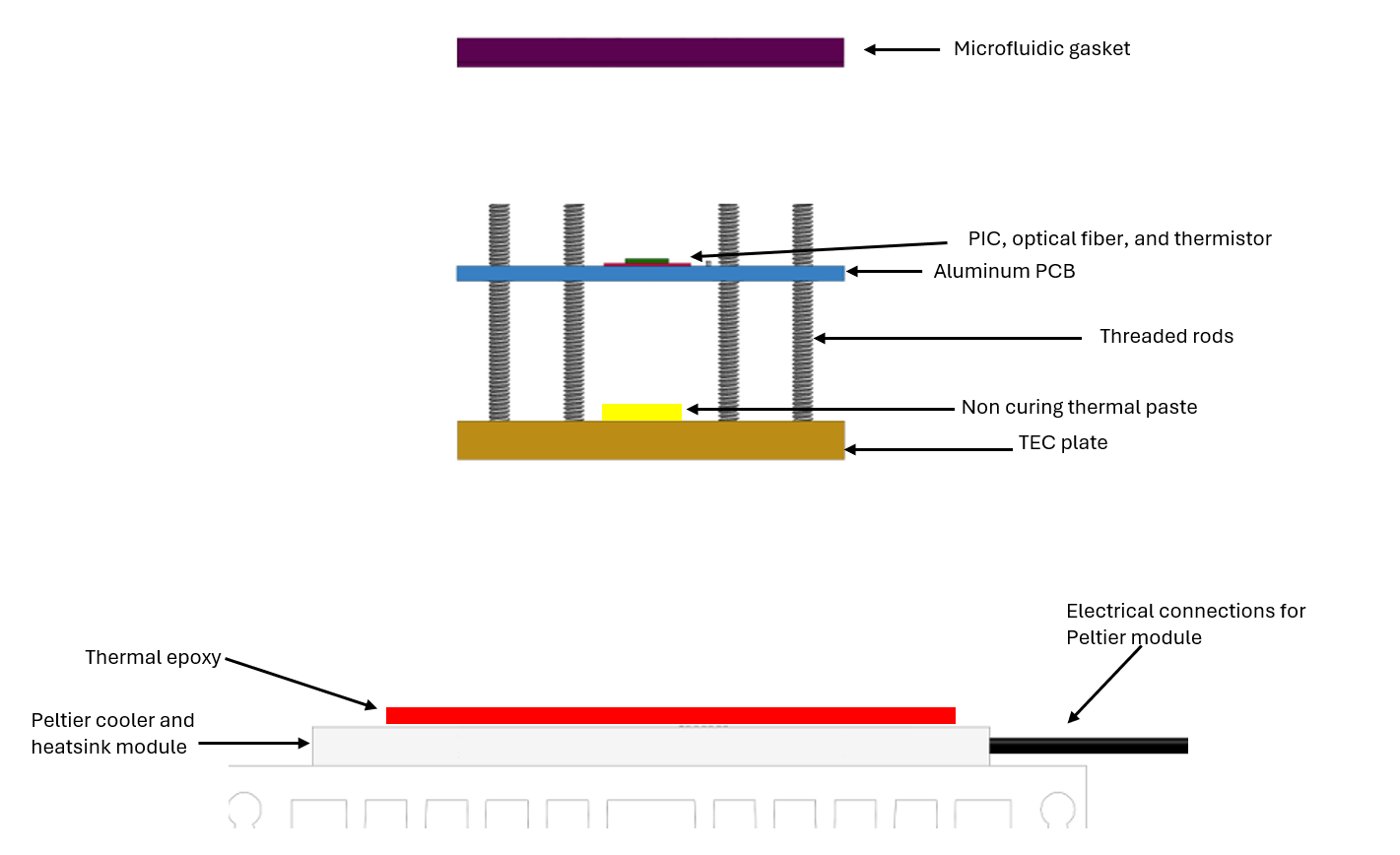


*Figure 4.1: PIC Mount System Diagram*

The overall assembly consists of four components:

1. A TEC module (white in Figure 5.1)
2. An aluminum plate tapped for threaded rods that we will call the TEC plate (yellow in Figure 5.1)
3. An aluminum PCB that will serve as a robust mount for each photonic chip and its fiber optic connections (light blue in Figure 5.1, with the photonic chip and the fiber array in red and green respectively)
4. A microfluidic gasket, provided by our client (shown in purple in Figure 5.1)

A variety of the design choices are motivated by the goal of achieving thermal control that is as precise as possible (GS-6). All components in this assembly are made of thermally conductive materials in order to reduce the thermal resistance between the chip and the TEC. Components are also made as thin as possible to reduce the overall thermal mass. A high thermal conductivity and low thermal mass reduces the thermal time constant and makes the system easier to control with the controller introduced in Section 2.3.2. A labelled stackup of the PIC Mount is shown in Figure 5.2.



*Figure 4.2: Side View of the PIC Mount*

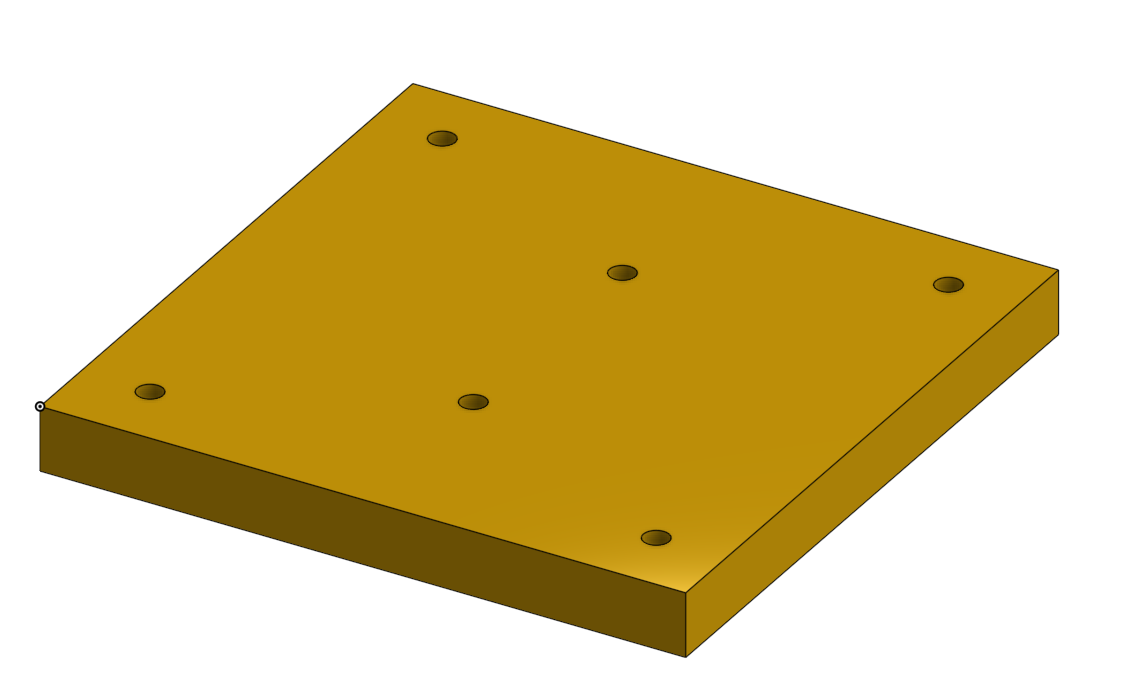
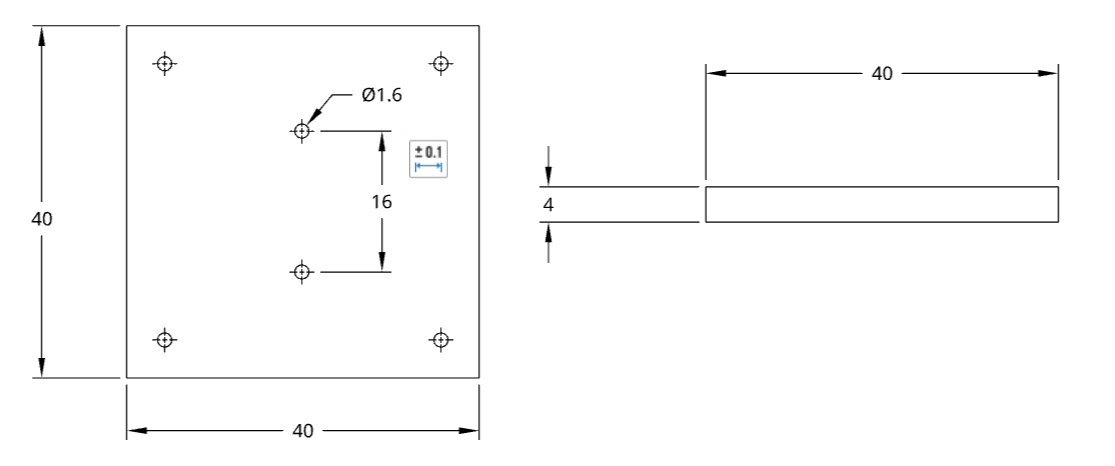
Good thermal contact is crucial to meeting the requirement of 5 mK control (RS-4). To ensure this, the TEC plate is attached to the TEC with thermal epoxy. The TEC plate is not meant to be replaceable, so a permanent, strong, and thermally conductive bond is required. The aluminum PCB is attached to the TEC plate with liquid, non-curing thermal paste to ensure sufficient thermal contact while achieving the required ease of swapability. The TEC plate has four threaded holes that align with cutouts in the aluminum PCB. The aluminum PCB is then held in place with nuts screwed down the threaded rods to interface robustly with the microfluidic gasket (one of our design constraints, CS-3). The microfluidic gasket is secured onto the photonic chip with a similar system consisting of two threaded rods and their respective nuts.

Each of the individual components of this system is listed below with justification for the design choices made.

## 4.1 TEC Module

An Adafruit TEC module and heat sink are used. The TEC is the main actuator for controlling the temperature of the photonic chip. It is driven by the control hardware on the eval PCB. It electrically pumps heat through itself to force heat into/out of the PIC stage. This component is chosen because it is preassembled, low cost, can sink away sufficient thermal energy, and utilizes a popular and well documented TEC module (TEC1-12706). It comes with a fan which the eval PCB board supplies power to. It has a maximum power draw of 60W which is more than sufficient for cooling our 9x9mm PIC.

## 4.2 TEC Plate



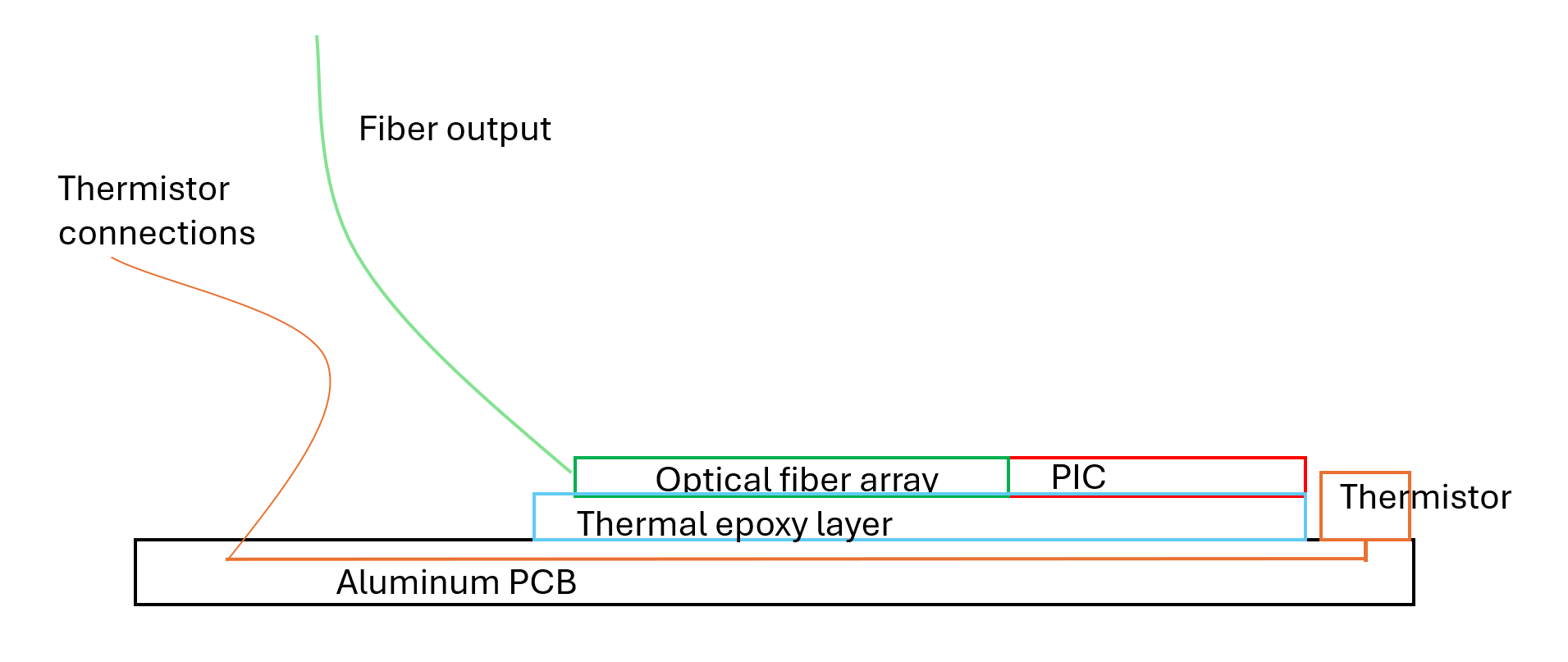
*Figure 4.3 - TEC Plate Drawings and 3D Model (dimensions in mm)*

This is the plate that is attached to the TEC directly with permanent thermal epoxy. It serves as an interface layer to tap holes into and attach threaded rods to interface with the microfluidic gasket (CS-3) as the TEC cannot be screwed into directly. There are 6 tapped holes, 4 for the rods that secure the aluminum PCB onto the TEC plate, and 2 to secure the gasket onto the aluminum PCB and photonic chip. Figure 5.3 shows the TEC Plate 2D schematics and 3D models.

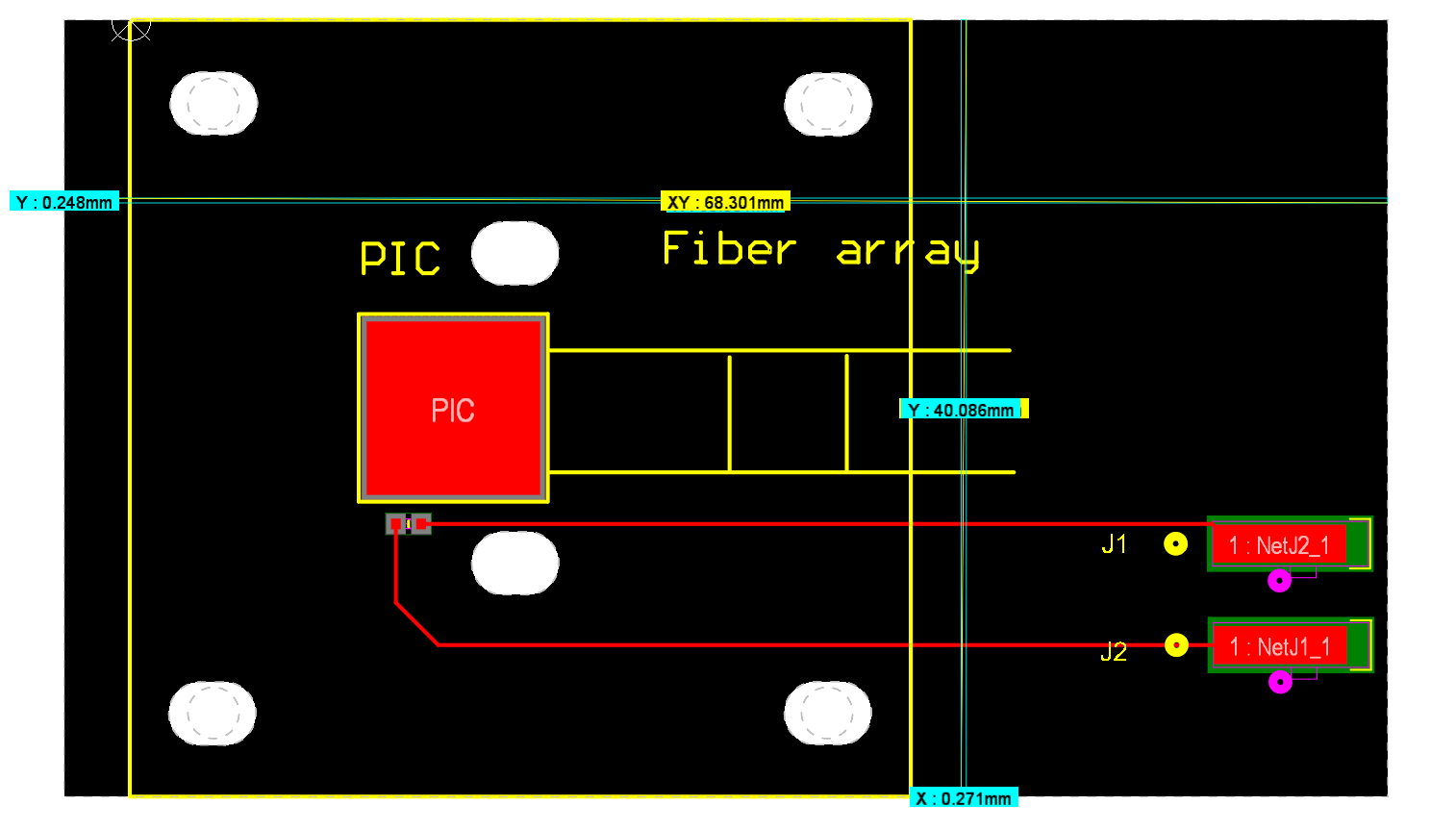
The location of the two middle threaded rods is chosen to match the client’s gasket dimensions. This plate is 4cm by 4cm (to match the dimensions of our TEC) and is 4mm thick. The holes are pilot holes which are machined out and then tapped for 2-56 threaded rods. The diameter of the threaded rod is chosen to match the size of the cutout in the gasket, and the thread spacing is chosen to ensure that enough threads are tapped into the 4mm thin plate for mechanical robustness (RS-9).

## 4.3 Aluminum PCB

An aluminum PCB is used as a hotswappable, thermally conductive platform for attaching the photonic chip and fiber array. The photonic chip and the fibers are attached to the aluminum PCB in a cleanroom by our client. This aluminum PCB is a swappable element that contains a given biosensor chip on it. This aluminum PCB sits on top of the TEC plate with non-curing thermal paste, the stackup is shown in Figure 5.4, and the layout of the aluminum PCB is shown in Figure 5.5.



*Figure 4.4 - Side View of Aluminum PCB*

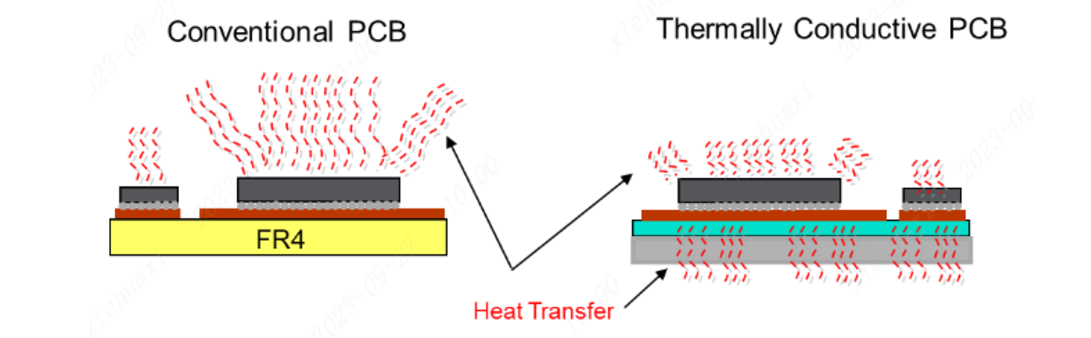


*Figure 4.5 - Aluminum PCB Layout*

The aluminum PCB is attached onto the TEC plate with the very same threaded rods and nuts used for the microfluidics. A thermistor is soldered onto the PCB very close to the photonic chip to provide good feedback for the temperature control system.

There are parallel lines centered on the photonic chip extending to the right as shown in Figure 5.4. They can be used as alignment markers for attaching the optical fiber array by the client. The photonic chip itself can be aligned with the yellow square labelled on the PCB as well as with a copper 9x9mm pad directly underneath it. This is to facilitate thermal contact with the PCB and to assist in alignment for the client working in the cleanroom.

Single layer aluminum PCBs are cost effective since no more than one electrically conductive layer is needed to route the thermistor. As shown in Figure 5.6, Aluminum PCBs have superior heat transfer to objects underneath them than a conventional FR4 PCB which is necessary here as the PCB sits on a TEC module.



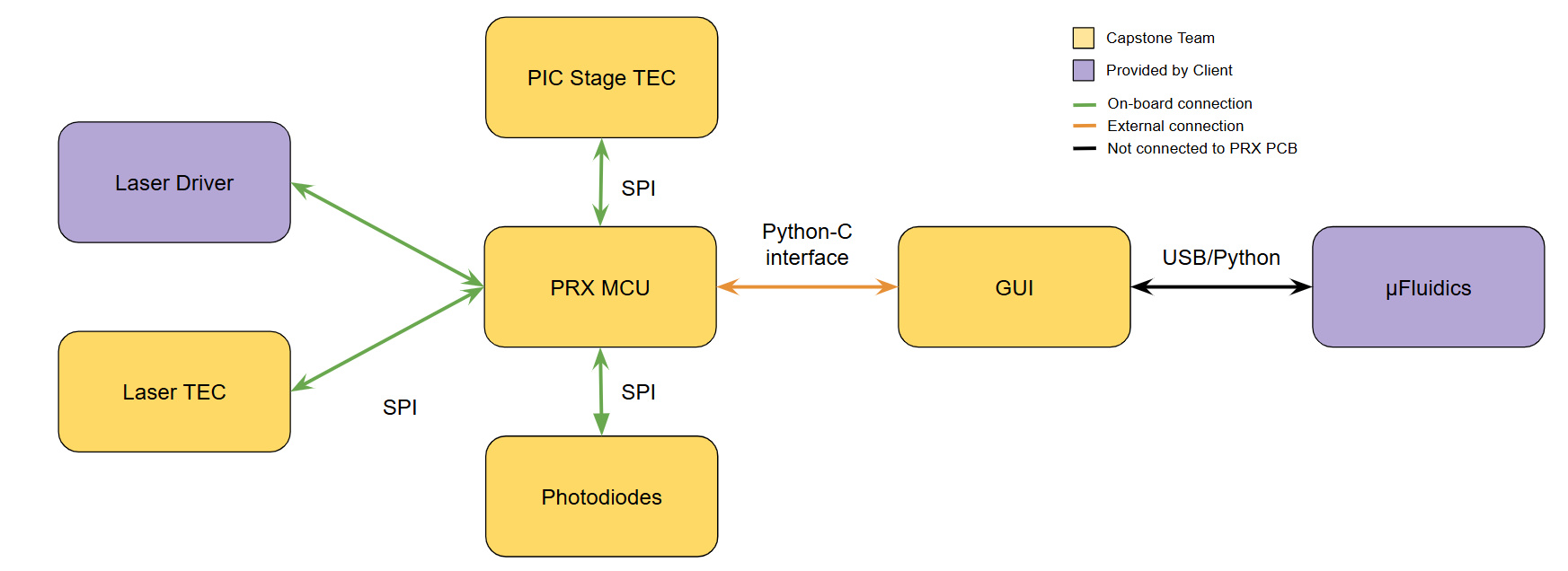
*Figure 4.6 - FR4 vs Aluminum PCB (Courtesy of JLCPCB [4])*

The thermistor chosen is NTCG104ED104DT1X by TDK technologies. This is due to its relatively high sensitivity, indicated by a high B coefficient value of 4250K at the intended operating temperature (25°C). It also has a small footprint (0402) which is necessary as it sits under the microfluidic gasket. Lastly, it has very precise resistance and B coefficient values with a tolerance of 0.5%. This allows us to control the temperature of the system to <5mK (RS-4).

The thermistor connects to the eval PCB using shielded 24 AWG wires (Tensility 30-00345). Shielded wires are chosen as the thermistor signal is susceptible to noise due to it being small in magnitude.

# 5 Firmware Low-level Design

The firmware on the STM32 communicates, controls, and synchronizes all devices in the system. It provides functionality for a user to monitor, acquire data, and configure settings on the evaluation system with a host PC. Figure 6.1 shows the subsystems interfaced by the firmware.



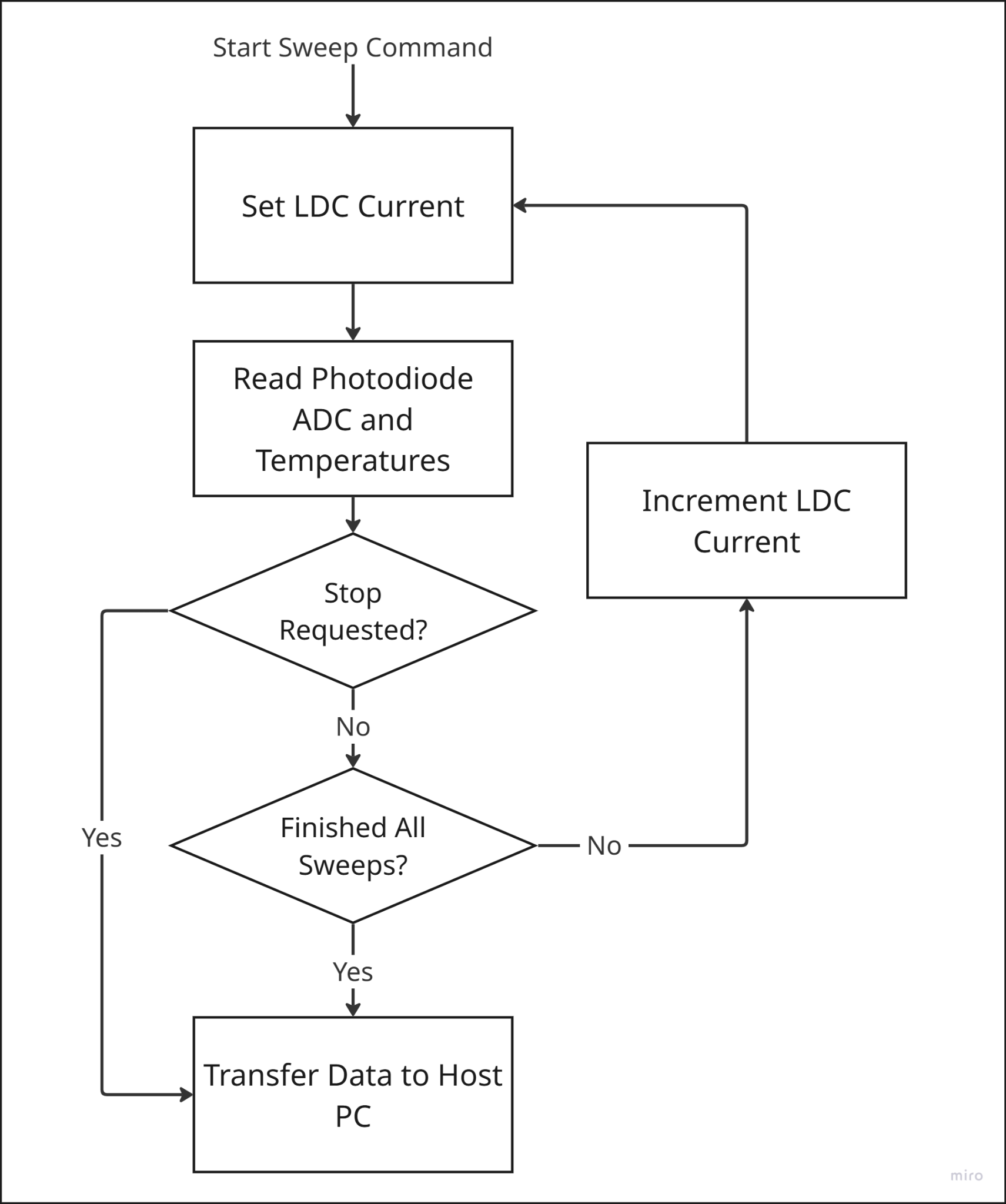
*Figure 5.1: Overview of firmware system*

The firmware is designed in a modular fashion, with many symbolic constants, structs, enums and helper functions used to make modifying a device’s configuration values, or adding commands to the PC interface quick and easy, even for somebody who has never seen the codebase before. This is an important feature because the system will be used and developed further by members of the SoC, and biosensing labs.

## 5.1 Laser and Photonic Chip Interface

In order to measure the resonant response of each photonic chip, the wavelength of the laser must be stepped over a range of wavelengths, with the photodiodes measuring the light power transmitted through the chip at each step. Wavelength can be changed by changing the driving current of the laser. The relationship between current and wavelength is roughly linear over the wavelengths of interest to our system. Synchronizing laser current setting, and reading photodiodes is done through the PC interface, and purely in firmware. The firmware synchronization is preferred for measurements since it is much faster, but the software synchronization can also accommodate benchtop equipment.

Autonomous sweeps carried out purely in firmware offer the benefit of high speed, eliminating the overhead of additional UART commands, and reducing the effect of environmental changes over the sweep period. The process of autonomously sweeping is shown in Figure 6.2. Software commands are necessary for interfacing parts of the evaluation system with other benchtop equipment, where external software is coordinating multiple instruments.



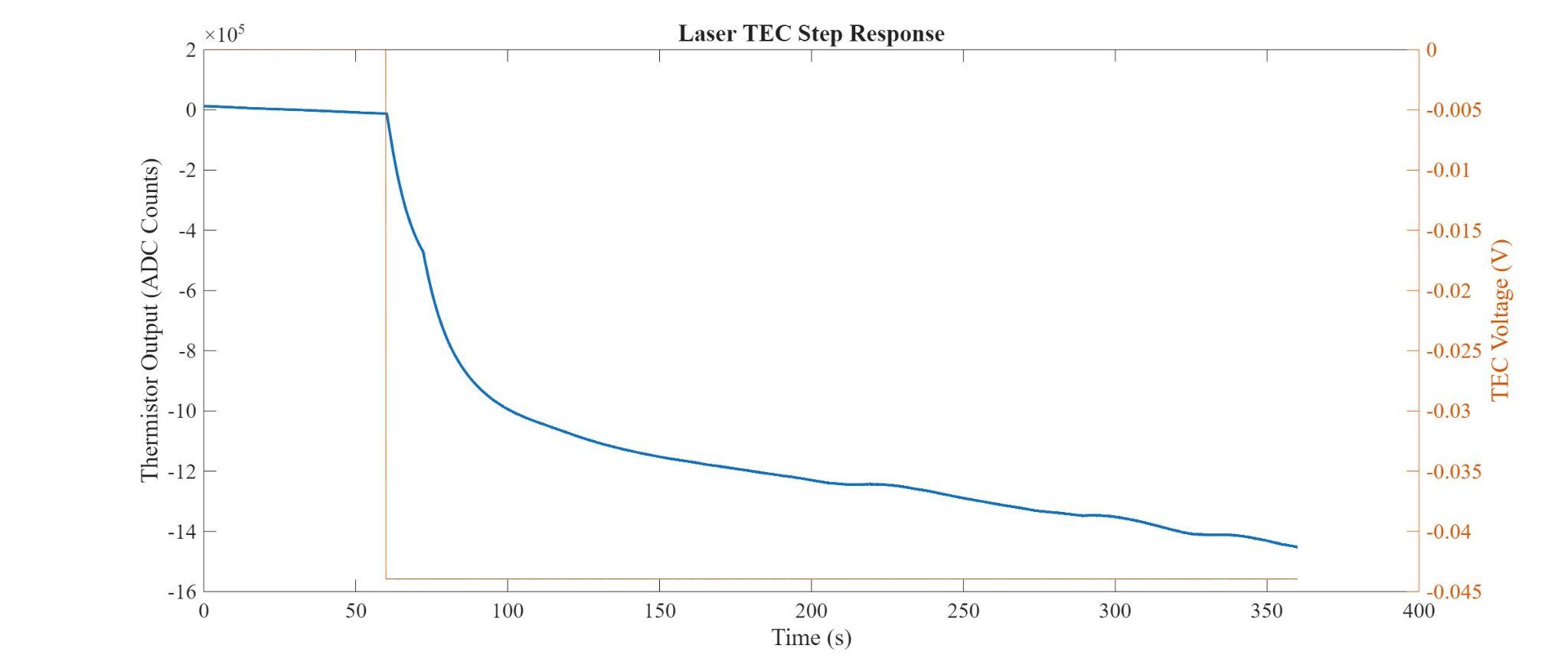
*Figure 5.2: Autonomous Sweep Flowchart*

### 5.1.1 Laser TEC Control

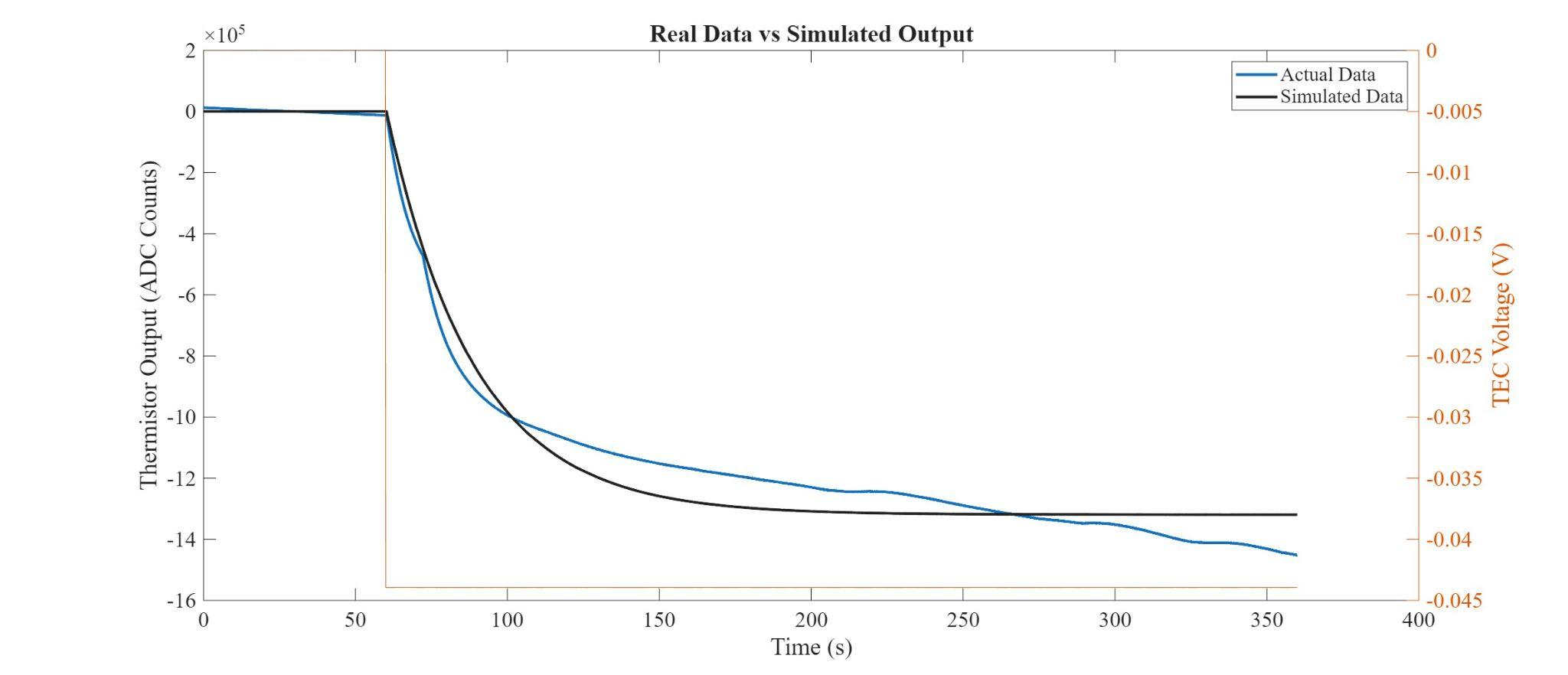
The laser is very sensitive to temperature changes, which requires it to be kept at a specific temperature during the entire sweep to ensure that the wavelength of the laser is only affected by the input current change and not the temperature fluctuations of the laser package, following requirement RS-4 (robust thermal control). This is especially difficult because as the laser current is increased during a sweep the laser package heats up which means that the controller has to reject a substantial temperature drift during the sweep.

The first step for PID controller design is to obtain a laser temperature - TEC voltage transfer function. The system is expected to be first order as it is a thermal system where the heat flow between the TEC and the laser diode depends on the temperature difference between the TEC and the Laser package containing the thermistor (ignoring the effect of any external disturbances).

A step response experiment is performed where the laser’s TEC potential is changed from 0V to -0.0440V (the negative sign indicating the TEC is cooling) and the laser’s temperature response is sampled at 9 Hz and reported for 5 minutes to give it time to stabilize and as the time constant of the system is expected to be on the order of one minute. Please note that the ADC output and temperature measurements are handled exclusively in ADC counts instead of voltages to reduce the CPU overhead of converting the ADC counts to voltage every clock cycle and as the relationship between ADC input voltage and output counts is linear, which allows the controller to act to stabilize the ADC counts instead of the measured voltage. The results of this are shown in Fig 5.3*.* As seen, the system response strongly mimics a first order system response which is expected.



*Figure 5.3. Blue Line - Laser TEC output (in ADC Counts, offset removed), corresponding to the left y-axis. Orange Line - Laser TEC Voltage (in Volts)*

Next, a model structure is chosen. An ARMAX (AutoRegressive Moving Average with eXogenous input) system model is chosen. This model structure is chosen as it is a standard discrete-time system model structure for controller synthesis and it includes a disturbance model which allows the model identification search to converge to more accurate models [5]. The System Identification Toolbox (Matlab, Mathworks) is used to obtain the system model. The output of the system identification process is shown in Fig 5.4.

*Figure 5.4: Black Line: Model Output. Blue Line: Real Data*

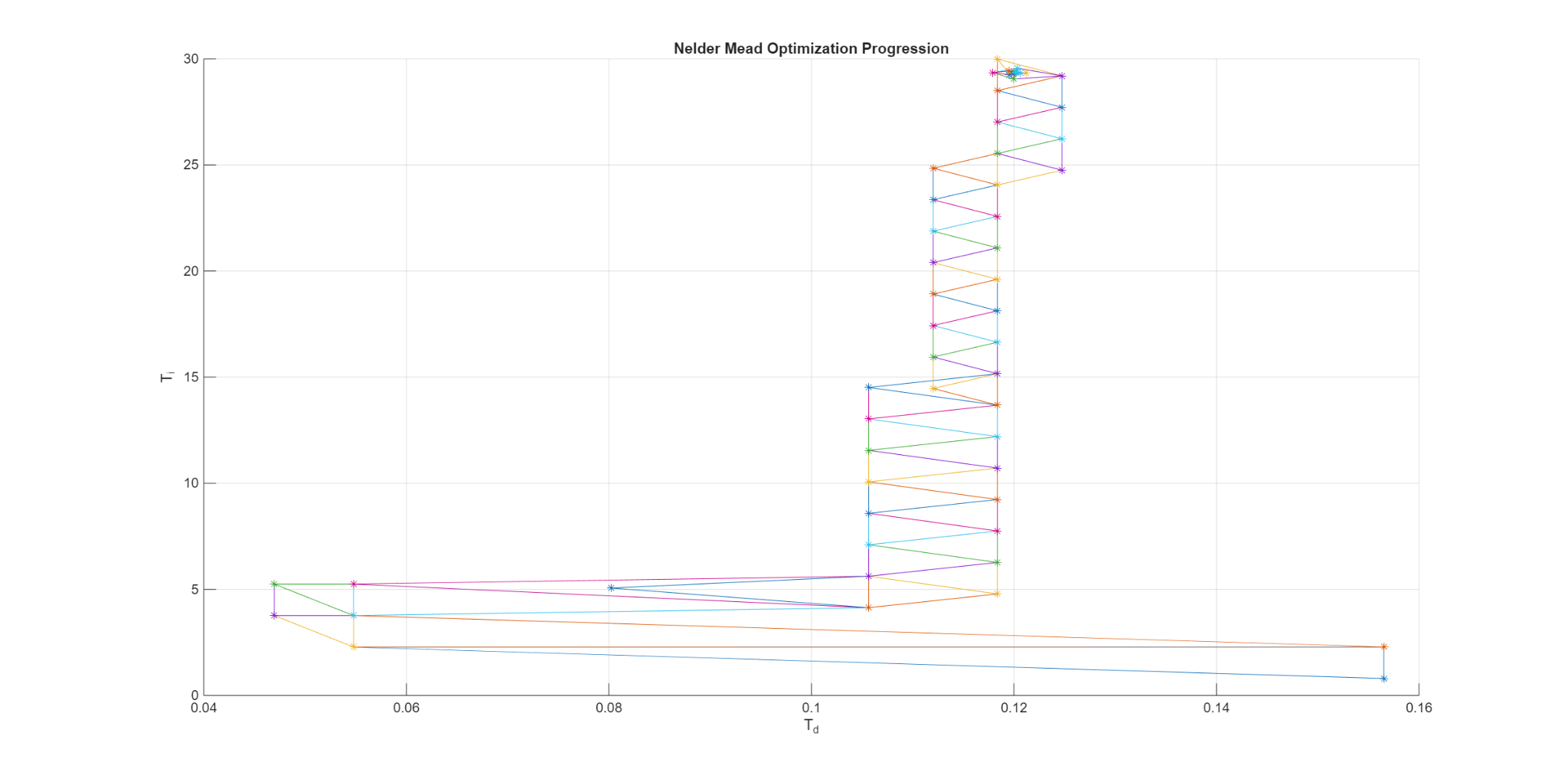
A fit of 86.13% is obtained which is sufficient for PID controller purposes (especially since most of the error comes from DC gain offset which isn’t as necessary for satisfying robustness constraints on the controller [5]).

A frequency of 200Hz is chosen for the PID, with 10x oversampling on the ADC to increase the temperature resolution, which increases the quality of the controller (GS-6). This requires resampling the obtained discrete time model to 200Hz. The high control frequency is chosen, despite the slow time constants of the system, as the controller’s main task is to reject high frequency disturbances as opposed to set-point tracking and so a sampling and control frequency fast enough to capture all the disturbance dynamics is necessary here.

The controller design method chosen is a constrained optimization of the open loop sensitivity function, described in further detail here [6]. This is chosen as it is a relatively modern and highly effective PID tuning method for systems where robustness is essential and good disturbance rejection (corresponding to temperature drifts in our case) is required, meeting design requirement RS-7 (Robustness) and goal GS-6 (accurate temperature control). In summary, the optimization problem is to recover from load disturbances (corresponding to laser temperature drifts or fluctuations in this case) with a minimal IAE (Integrated Absolute Error) of the error signal (the difference between the setpoint and the current sensor reading). The IAE is minimised within the constraint that the maximum sensitivity (Ms), the peak of the transfer function relating disturbances and controller response, is kept less than or equal to a chosen value. A higher allowed maximum sensitivity means that the controller can reject disturbances faster and that the closed loop bandwidth of the system is higher, at the cost of reduced phase margin and increased risk of instability due to system model errors. A lower maximum sensitivity means that the controller is more robust and tolerant of system model errors, at the cost of a lower closed loop system bandwidth and slower disturbance rejection. [6]. Since the system model is quite accurate and incredibly fast disturbance rejection is needed here, an Ms value of 2.2 is chosen.

A modified version of the SWORD PID design tool [7] is used for the constrained optimization search. It uses a Nelder-Mead optimization method over the space of all controllers that satisfy a user defined robustness constraint to design and obtain a filtered PID controller. The filter parameters were chosen following modern PID controller filter design research [8]. The main modification made to the SWORD tool is changing the controller search space from Continuous Time PID to Discrete Time PID controllers (using the Bilinear Transform, as that gives the best frequency mapping between continuous and discrete time systems [9]) to allow us to use more aggressive sensitivity margins without fear of running into instability.

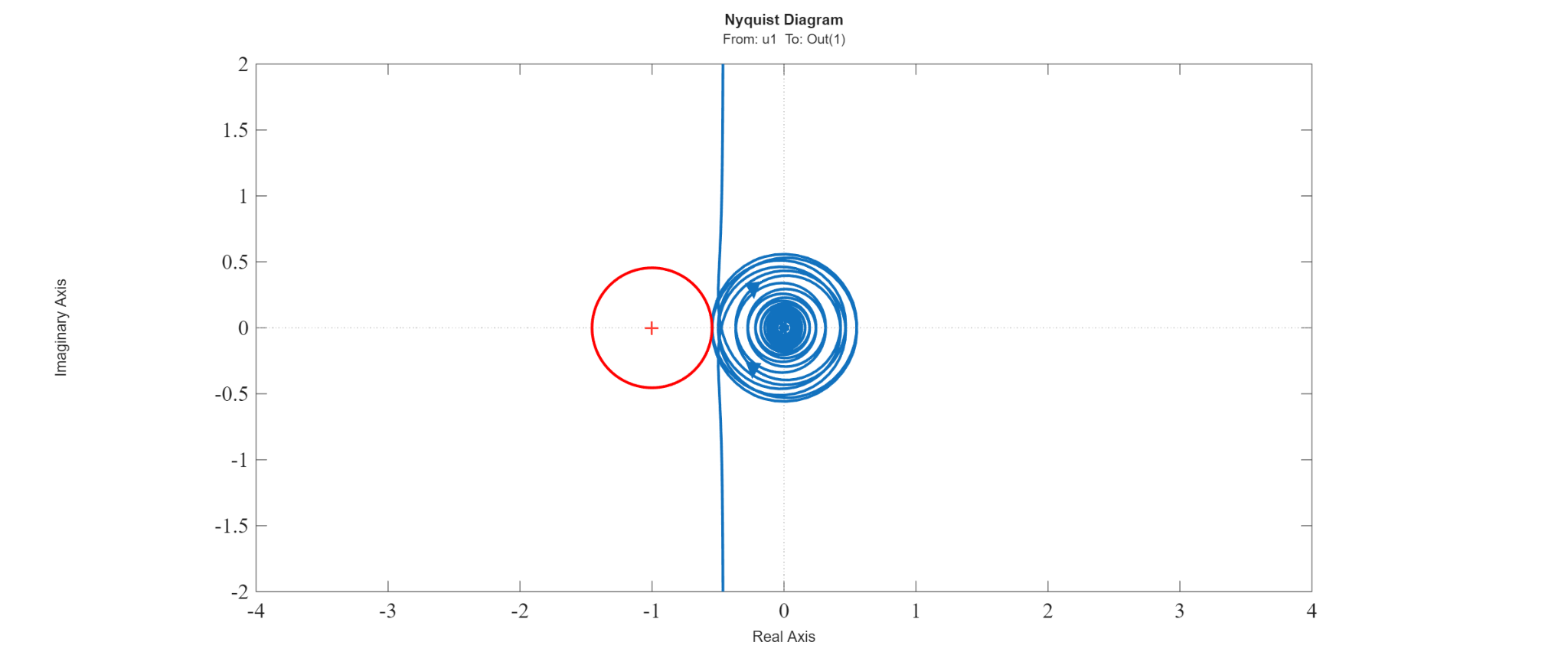
The search algorithm’s progress and output is shown in Fig 5.4



*Figure 5.5: The output of the search algorithm for a PID Controller.*

Each point corresponds to a different PID controller tested, with the algorithm converging in 53 iterations on a Ti of 29 and a Td of 0.12, where Ti and Td are the Integral time and Derivative time respectively

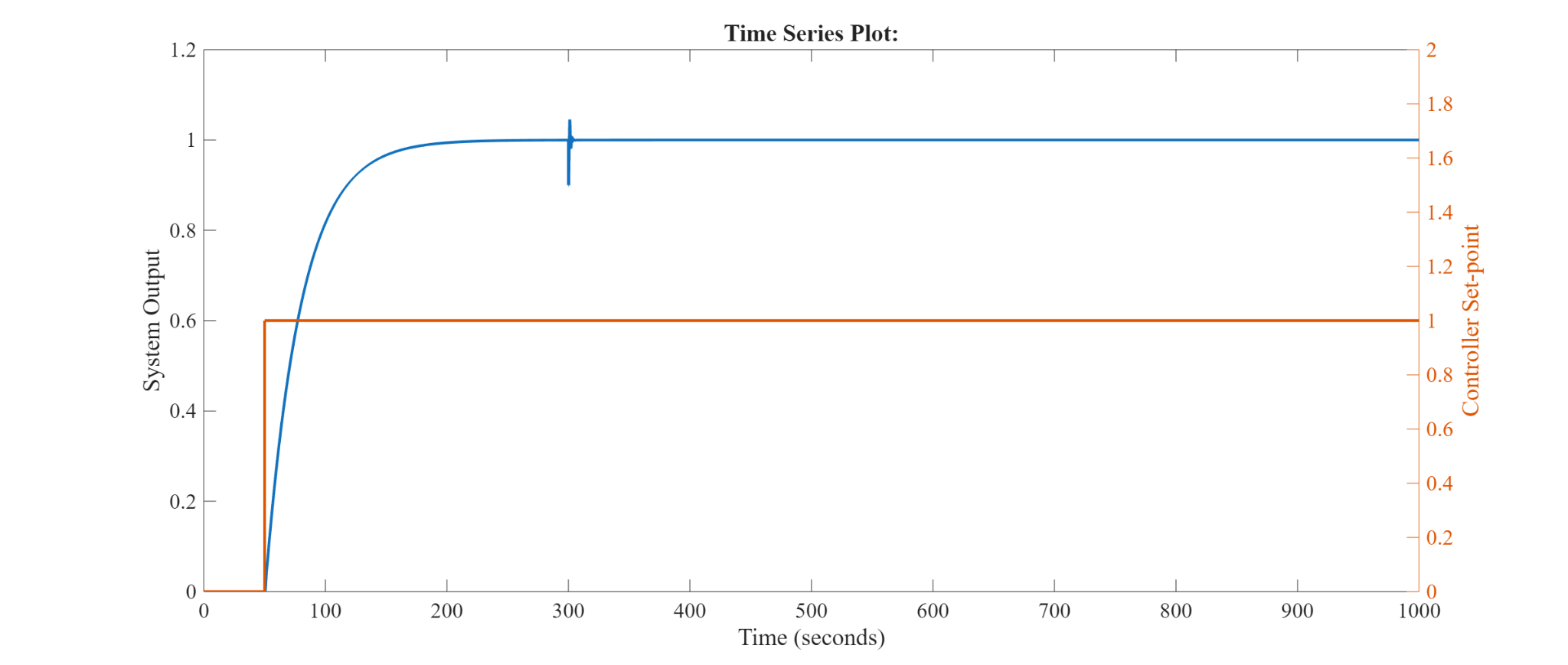
This controller results in the open loop system Nyquist plot shown in Fig 5.5



*Figure 5.6: Resulting Open Loop Nyquist Plot from PID Design tool. The red circle is the sensitivity constraint (corresponding to all points on the Nyquist plot with a maximum sensitivity of 2.2) and the blue line is the resulting open-loop system Nyquist plot.*

A 2 Degree-Of-Freedom (DOF) PID architecture is chosen where the derivative term only acts on the measured output instead of the error signal to reduce the “derivative kick” effect commonly seen with PID controllers [6].

The controller is next simulated for its set-point response and rejection of a change in the thermistor reading (corresponding to the laser package’s temperature drifting). The results of that are shown in Fig 5.6 and are satisfactory to move on to implementing the controller physically.



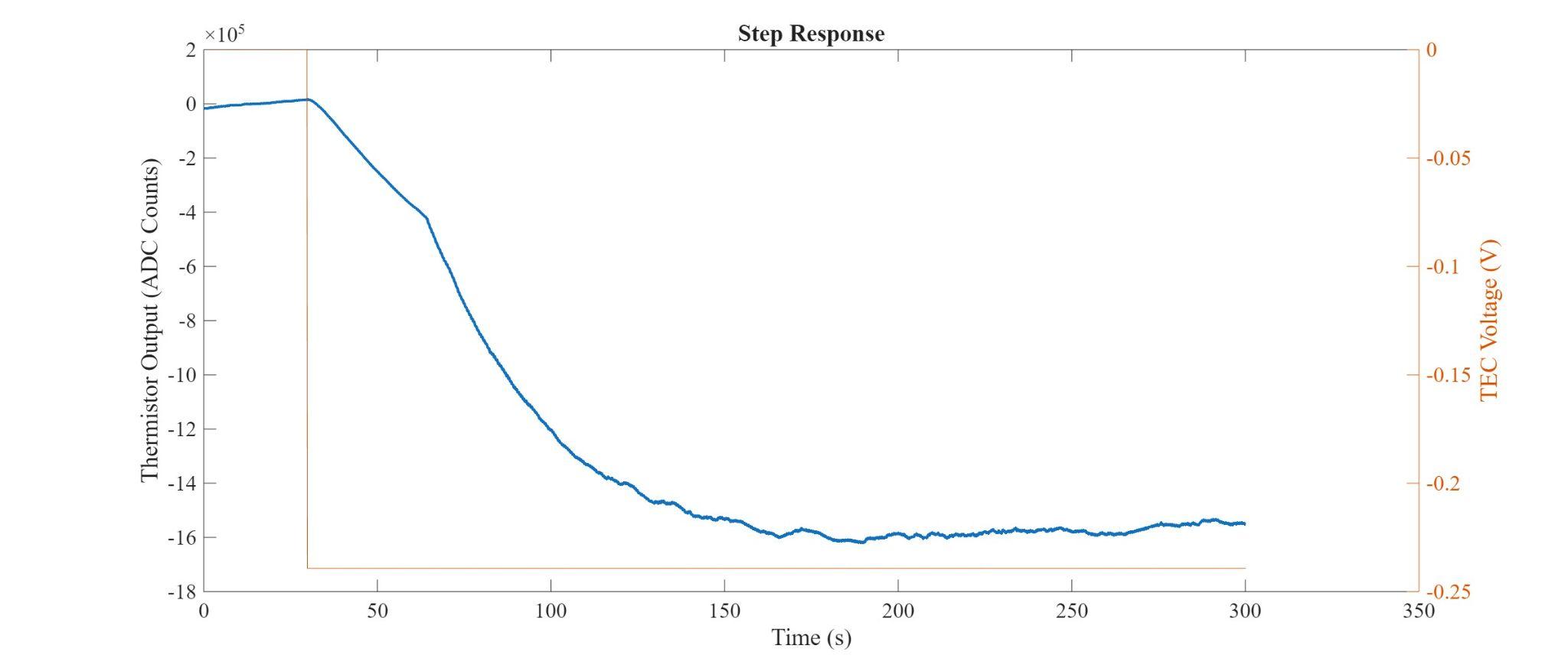
*Figure 5.7: Orange Line (right y-axis) - Setpoint of the controller. Blue Line (left y-axis), laser temperature (offset removed). At t = 300s, a disturbance corresponding to 10% of the laser’s current temperature is introduced to the output to monitor the controller’s response to it.*

Lastly, the resulting filter-controller series connection is transformed from a Z-domain transfer function into a difference equation to be implemented on the microcontroller in C. The results of testing and implementing the controller are discussed in the V&V document.

### 5.1.2 Stage TEC Control

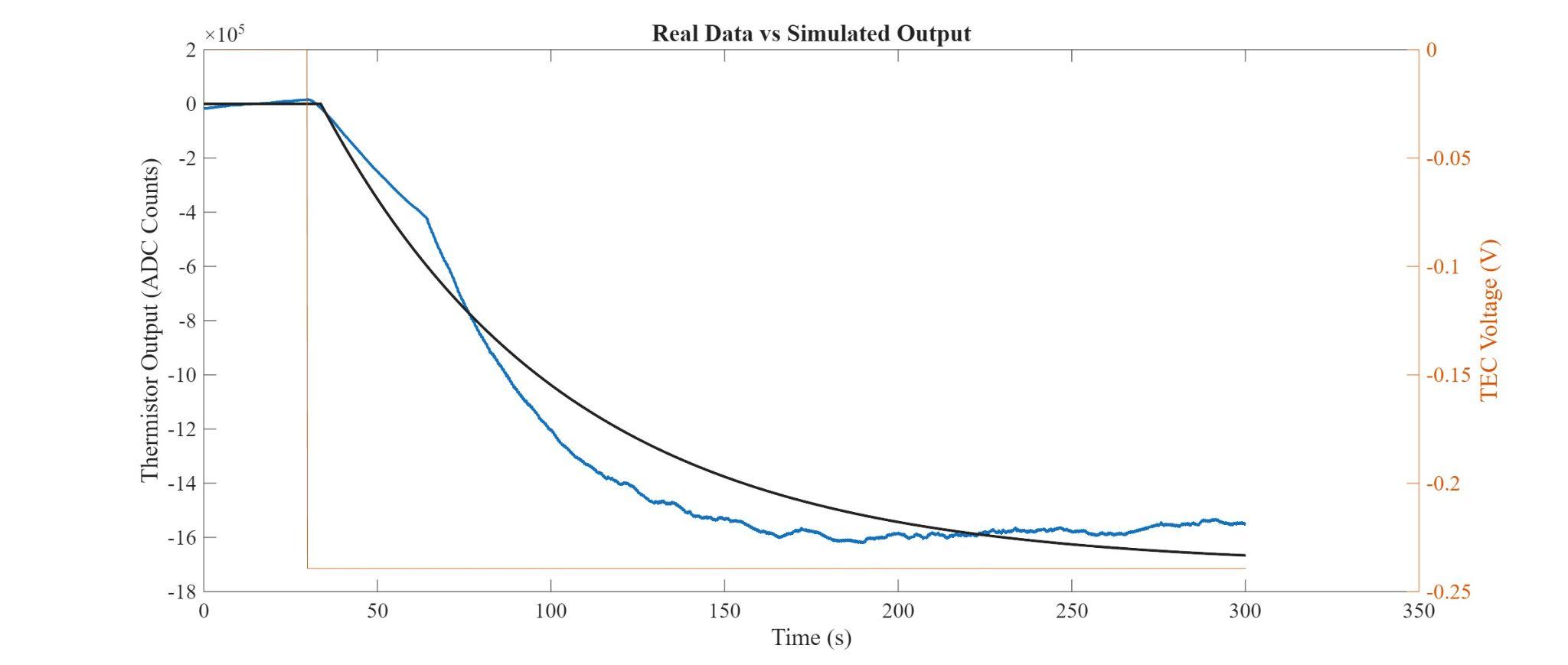
Congruent to RS-4 (thermal control), the stage and the DUT (Device Under Test) needs their temperature controlled to within 5mK of a stable, fixed set-point. Similar to the Laser TEC Control section described above (5.1.1), the first step of the controller design is to obtain a system model to design the PID controller for. The system is again expected to be first order as it is a thermal system where the heat flow between the TEC and the stage depends on the temperature difference between the TEC and the TEC plate containing the thermistor (see section 4.2 for more details on the TEC plate), ignoring the effect of disturbances.

A step response experiment is performed where the stage TEC’s potential is changed from 0V to -0.239V (the negative sign indicating that the TEC is cooling) and the stage’s temperature response is sampled at 9 Hz and reported for 5 minutes to give it time to stabilize and as the time constant of the system is expected to be on the order of one minute. Please note that the ADC output and temperature measurements are handled exclusively in ADC counts instead of voltages to reduce the CPU overhead of converting the ADC counts to voltage every clock cycle and as the relationship between ADC input voltage and output counts is linear, which allows the controller to act to stabilize the ADC counts instead of the measured voltage. The results of this are shown in Fig 5.7*.* As seen, the system response strongly mimics a first order system response which is expected.



*Figure 5.7: Blue Line - Laser TEC output (in ADC Counts, offset removed), corresponding to the left y-axis. Orange Line - Laser TEC Voltage (in Volts)*

Next, a model structure is chosen. Again, an ARMAX (AutoRegressive Moving Average with eXogenous input) system model is chosen. This model structure is chosen as it is a standard discrete-time system model structure for controller synthesis and it includes a disturbance model which allows the model identification search to converge to more accurate models [5]. The System Identification Toolbox (Matlab, Mathworks) is used to obtain the system model. The output of the system identification process is shown in Fig 5.8.

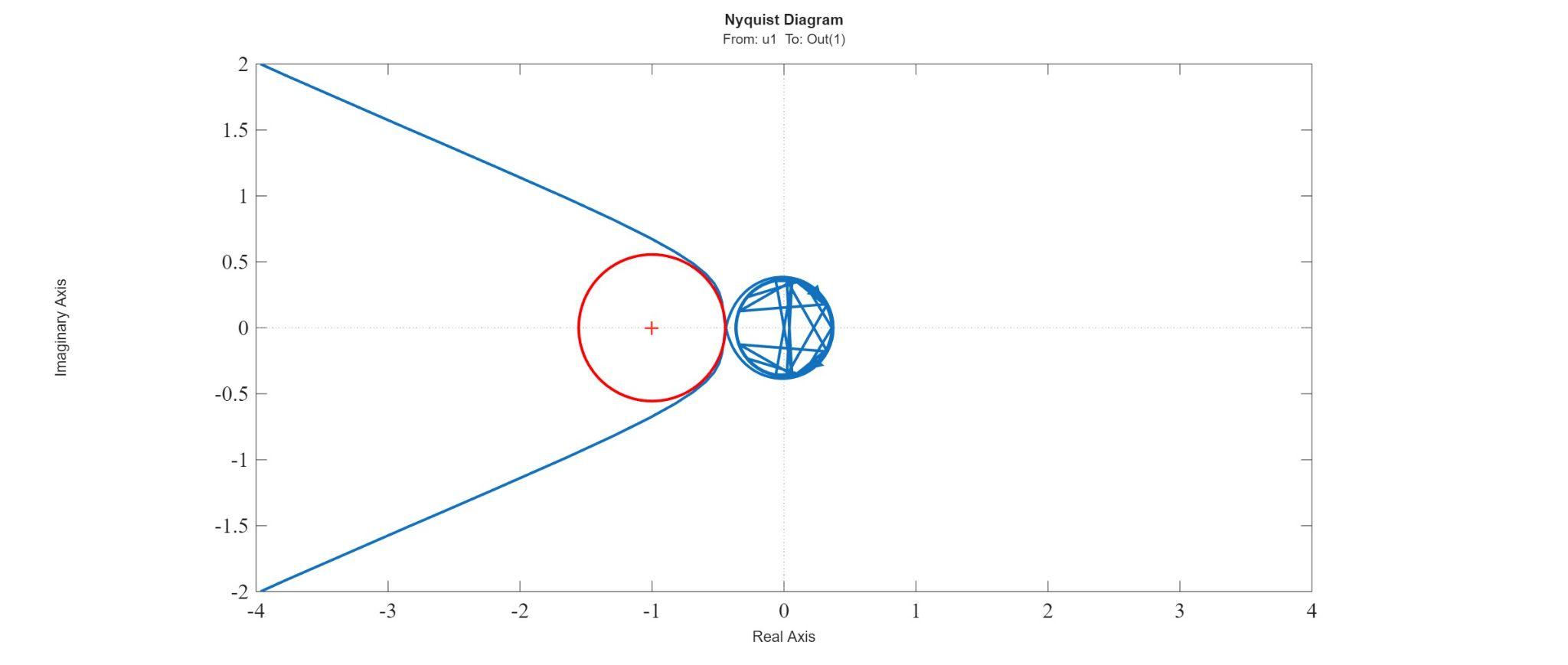


*Figure 5.8: Black Line: Model Output. Blue Line: Real Data*

A fit of 81.41% is obtained which is sufficient for PID controller purposes (especially since most of the error comes from DC gain offset which isn’t as necessary for satisfying robustness constraints on the controller [5]).

A frequency of 200Hz is chosen for the PID, with 10x oversampling on the ADC to increase the temperature resolution, which increases the quality of the controller (GS-6). This requires resampling the obtained discrete time model to 200Hz. The high control frequency is chosen, despite the slow time constants of the system, as the controller’s main task is to reject high frequency disturbances as opposed to set-point tracking and so a sampling and control frequency fast enough to capture all the disturbance dynamics is necessary here.

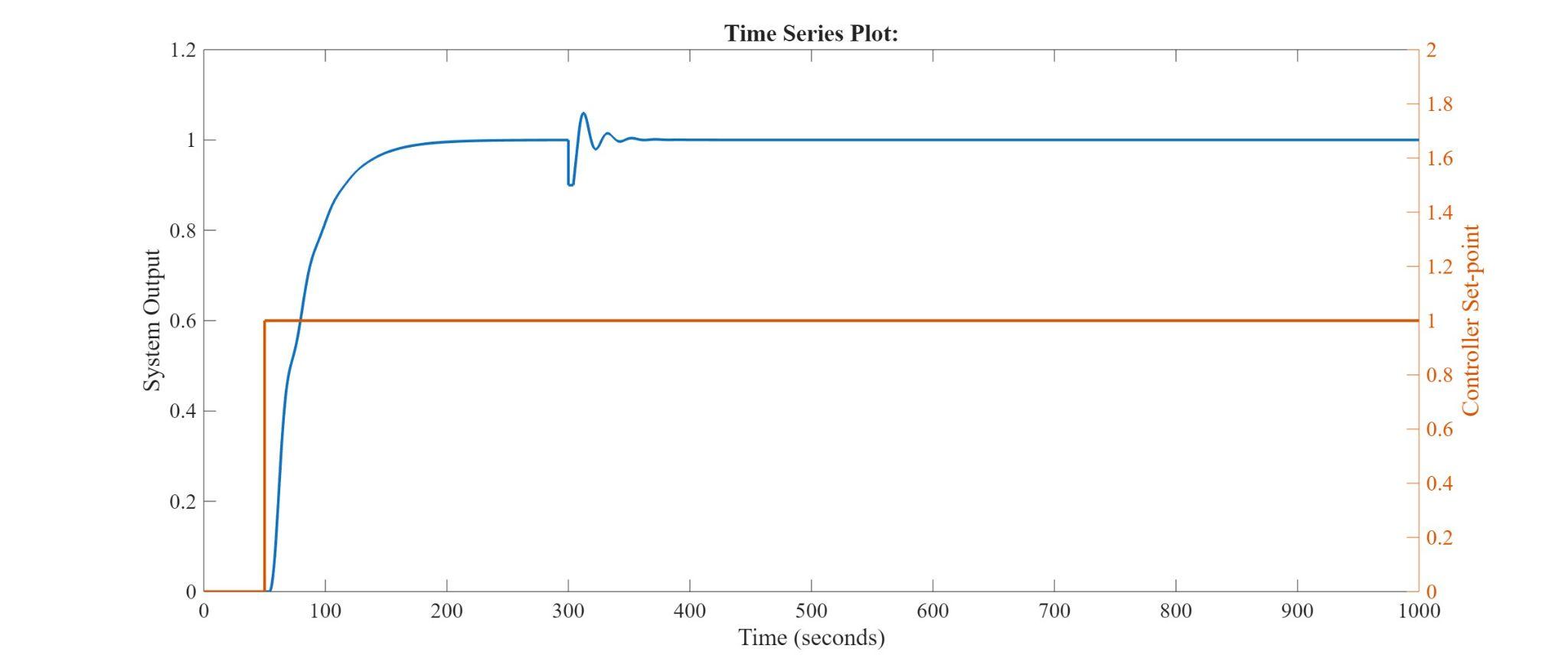
For the controller design, unlike with section 5.1.1 where the system’s temperature is strongly drifting, since the TEC is a much larger and more powerful heat source/cooling device than the PIC/Fiber Fabry-Perot interferometer, less aggressive tuning methods can be used. Therefore, constrained convex optimization using the matlab tool CVX, a package for solving convex problems [10],[11]. This was done following the approach described in [12], where a controller that satisfies stability requirements while maximising load disturbance rejection (caused in this case by the Stage’s temperature changing or the PIC heating up) is iteratively solved for. This is subject to a maximum sensitivity constraint again. Since the system model fit is a little lower than for the Laser (section 5.1.1), a lower maximum sensitivity of 1.8 is chosen. The resulting open loop Nyquist plot is shown in Fig 5.9



*Figure 5.9: Resulting Open Loop Nyquist Plot from CVX. The red circle is the sensitivity constraint (corresponding to all points on the Nyquist plot with a maximum sensitivity of 1.8) and the blue line is the resulting open-loop system Nyquist plot.*

Lastly, this resulting controller is filter-free, which makes it difficult to implement in reality as the derivative term is especially sensitive to measurement noise. Therefore, a filter term of the structure is chosen where Td is the derivative time and K is the proportional gain of the controller [6]. N is a tunable parameter corresponding to the degree of filtering and is set to 10 in this case [6]. A 2 Degree-Of-Freedom PID structure is chosen again to mitigate the effect of the “derivative kick”, described in [6].

Lastly, the controller’s performance is evaluated in simulation (shown in Fig 5.10) and the controller is implemented by turning the Discrete-Time transfer function of the controller (obtained using the Bilinear Transform [9] from the continuous time controller transfer function) into a difference equation.



*Figure 5.10: Orange Line (right y-axis) - Setpoint of the controller. Blue Line (left y-axis), laser temperature (offset removed). At t = 300s, a disturbance corresponding to 10% of the stage’s current temperature is introduced to the output to monitor the controller’s response to it.*

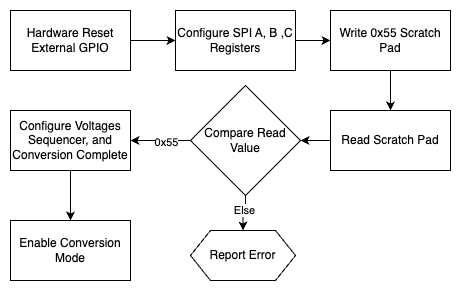
Note that the differences between the tuning choices here and the laser controller’s tuning choices are owing to the difficulty of controlling the laser and thus the necessity of more advanced tuning methods and of the entire design process being in discrete time. Since the Stage is a much less relaxed tuning problem, a faster, easier and more computationally efficient method is used.

## 5.2 Photodiode Interface

The firmware takes measurements of the photodiode power by triggering an ADC reading. The photodiode ADC is interfaced on its own dedicated SPI line because it supports faster communication than the other hardware interfaced over SPI. Faster communication allows the CPU to either improve temperature control (GS-6) or run sweeps faster (GS-1), both of which are design goals. This ADC has two modes: configuration and conversion. Conversion mode is when samples are actually taken, while configuration mode sets the details of the sampling.

### 5.2.1 Configuration Mode

The ADC has a lot of functionality, which can be configured by setting registers. In configuration mode, SPI commands can write to or read from the configuration registers. Figure 6.3 shows the ADC configuration routine. The ADC can perform oversampling, only returning the final result, which reduces CPU overhead on the MCU, allowing for faster sweeps and better temperature control. It is set to oversample at a rate of 64x. This gives enhanced resolution of the photodiode power, which helps resolve peaks more accurately, pursuant to our design goal (RS-2).



*Figure 5.11: ADC Setup Flowchart*

When the current steps, a read of all 16 photodiode channels is triggered. This ADC can be set to oversample at 1x,4x,16x or 64x. These correspond to a resolution of 16, 17, 18 and 19 bits respectively. The conversion pin is toggled for each sample before the result is output by the ADC at an increased resolution. This process is repeated 15 more times, to sample each photodiode. The ADC uses its built-in sequencer to automatically cycle to read the next photodiode after data for the current one has been acquired. The MCU knows when the data is ready by polling the BSY pin on the ADC, which is a GPIO that is set low when the data is ready. This data is immediately transmitted to the host computer using a custom bit packing algorithm, explained below. The ADC can support sampling up to 500KHz. It can also handle SPI communication up to 25MHz, but for prototyping purposes, the SPI communication is configured to run at 1.5MHz.

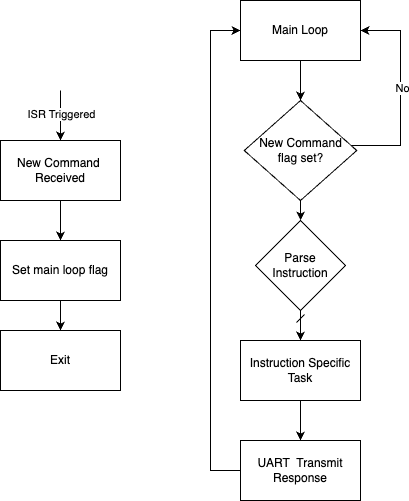
Since some of these resolutions do not neatly line up with byte boundaries, a custom bit packing and unpacking scheme is employed to minimize the amount of data sent. Since there are 16 photodiode channels, the overall number of bits that need to be transferred is 16\*resolution, so the number of bytes to be sent is 2\*resolution. Our scheme packs all 16 values contiguously into a buffer of this size then transmits the buffer byte by byte. The receiver then unpacks 2\*resolution bytes back into their original 16 readings at the specified resolution. This scheme saves between 20-30% in communication overhead, depending on the oversampling rate, compared to zero-padding, which directly translates to a faster sweep (GS-1).

### 5.2.2 Conversion Mode

When the current steps, a read of all 16 photodiode channels is triggered. The ADC is configured for 64x oversampling. The conversion pin is toggled for each sample before the averaged result is output by the ADC at an increased resolution. This process is repeated 15 more times, to sample each photodiode. The ADC uses its built-in sequencer to automatically cycle to read the next photodiode after data for the current one has been acquired. The MCU is informed the data is ready by polling the BSY pin on the ADC, which is a GPIO that is set low when the data is ready. This data is saved to memory to be transmitted to the host PC at the end of the sweep. The ADC can support sampling up to 500KHz. It can also handle SPI communication up to 25MHz, but for prototyping purposes, the SPI communication is configured to run at 1.5MHz.

## 5.3 PC Interface

The communication protocol and message structure between devices can be seen in Appendix B. Most of the functionality is hidden from the PC, but a few key functions are exposed. This allows the GUI to easily send and receive commands, while keeping the software codebase as small and as simple as possible. As a result, this design choice helps to meet the requirements of integrated software control (RS-5) and software data acquisition (RS-6). The connection is established over a USB cable, where the serial messages transmitted by the PC are converted to UART for the MCU via a UART/USB bridge on the STM32 NUCLEO board. When a command is sent to the MCU, an interrupt is triggered, setting a flag for the main loop to parse and run the relevant code. Once completed, a response is sent to the PC indicating the completion along with any requested data. This process is shown in Figure 6.4. A full definition of message formats is given in the Appendix.



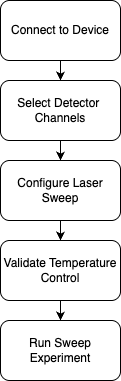
*Figure 5.12: Communication between MCU and GUI Flowchart*

# 6 Software Low-level Design

The software interfaces with the eval PCB. It provides the user an easy and intuitive platform to customize experiments, initiate sweeps, set PID parameters, check controller performance, and store their data.

## 6.1 GUI

The GUI is designed for the user to follow a linear configuration procedure, simplifying experiment set-up. The core steps are outlined in the following diagram.



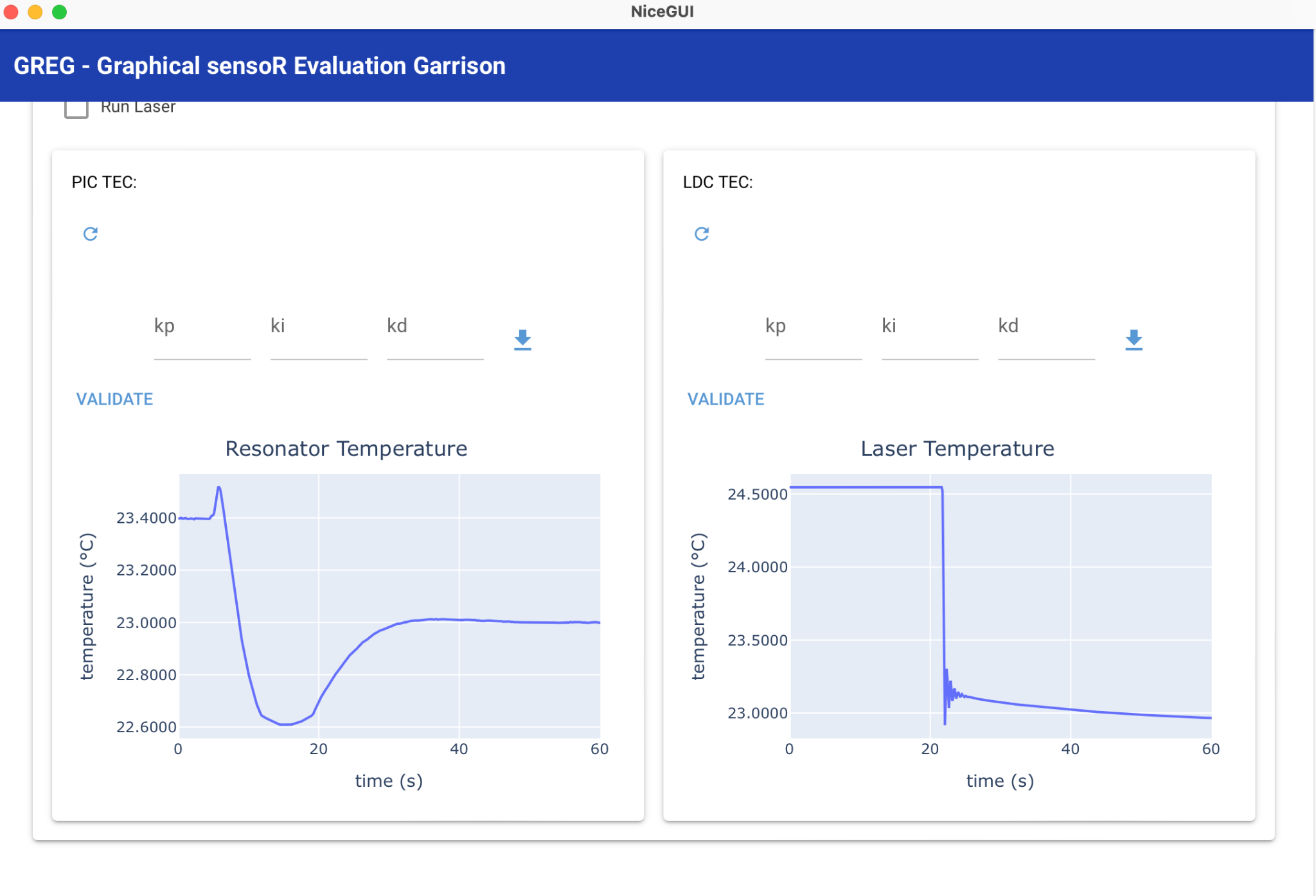
*Figure 6.1: GUI Configuration and Experiment Setup*

Device connection requires no additional input from the user as the software automatically detects the hardware if connected to the host computer. If the hardware becomes disconnected, this is also detected and notified to the user. Once connected, the user configures which detector channels are to be saved for the experiment. The laser sweep is configured by entering start, stop, and step current amounts, along with the number of times this sweep is to be performed. These values are then saved to the hardware and a linear current ramp is generated to drive the laser with.



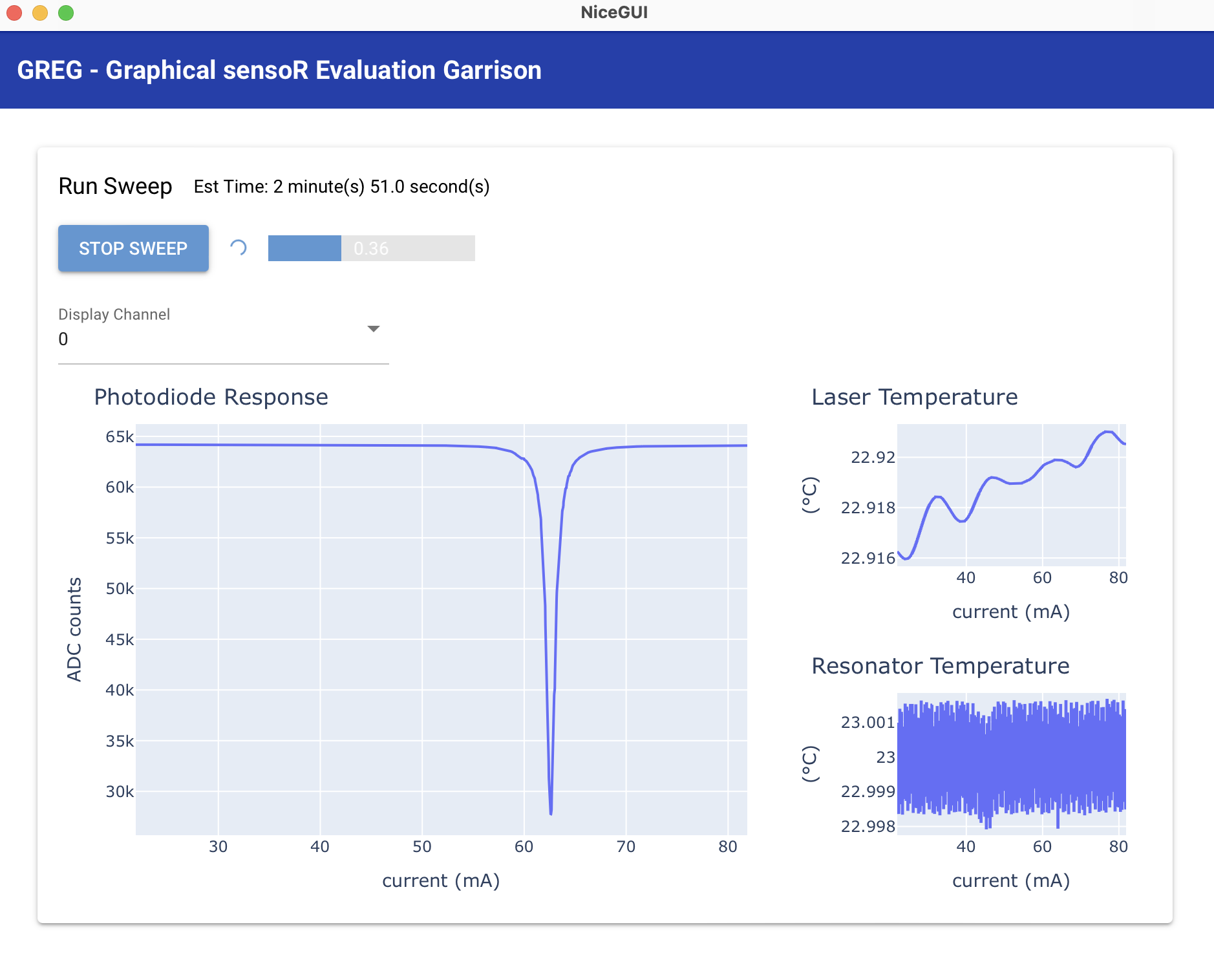
*Figure 6.2: Connecting a device and configuring a custom experiment*

Due to the strict temperature requirements, it is crucial to verify both the laser and resonator temperature have stabilized at their setpoints prior to starting the experiment. A dedicated tab for live plotting temperature along with adjusting PID parameters is provided, allowing the user to tune the response in real time. Because temperature control of the laser will behave differently under steady state and current sweeps, there is an option to run the configured current sweep while monitoring the temperature to accurately assess temperature control performance prior to starting the experiment.



*Figure 6.3: Verifying the temperature controller*

Once properly configured and temperature control has been validated, the user is ready to run the sweep experiment. An execution time estimate is provided, along with a live progress bar while running. During the sweep, the user can switch the plotted response between any of the configured photodiode channels. Along with the photodiode data, both the laser and resonator temperatures are displayed per sweep, verifying the temperature is within an acceptable range.



*Figure 6.4: Running a sweep*

While the sweep is running, the GUI is saving the incoming data to a .mat file for ease of analysis. Because the data for each sweep can be sent to the computer faster than the run time for a single sweep, data from sweep (n-1) is sent to the software while sweep (n) is running, allowing for the maximum throughput without waiting between sweeps. The file is written to and closed each time new data arrives, meaning that in the event of a crash or disconnect from the hardware, all data up to that point is safely saved rather than only saving data after completing the experiment.

All communication with the hardware is done asynchronously from the GUI, so the user can still move around and interact with the software while experiments and other communication are happening. This allows for a lightweight experience and the ability to monitor multiple device aspects at the same time. Features to prevent conflicting actions have been implemented, such as disabling attempts to change the sweep configuration while the experiment is running.

## 6.2 Backend

Communication with the host computer running the GUI, is done through a custom command interface designed by our team. The commands and their function can be found in Appendix B. Our team designed the firmware code to make this interface easily reconfigurable without a full understanding of the system. This was done by carefully labelling all configuration registers and their fields with symbolic constants, structs and enums. As a result, adding new functionality to the firmware is easy, quick, and can be done without looking at any datasheets.

## 6.3 Sweeps

The first kind of sweep is the fully autonomous sweep as shown in Figure 3.1. Fully autonomous sweeps can be performed by hardware integrated on the PCB without external instruments. The key resolution (RS-2) and repeatability (RS-3) requirements apply to these autonomous sweeps primarily. The software’s procedure for this sweep is as follows:

1. The software sends the MCU the starting point, ending point and step size of the desired laser current sweep
2. The software waits for the eval PCB to finish performing the requested sweep
3. The software receives the results from the eval PCB board over serial communication in a batch sent at the end of the sweep

Since this sweep is performed completely in firmware, it is much faster and helps meet the goal of high-speed measurement (GS-1).

**

*Figure 6.5: Autonomous sweep software flow diagram*

# 7 Conclusion

Using custom hardware, our system is able to effectively resolve the transmission curve of a photonic resonant cavity. Our team demonstrates this by using a fiber Fabry-Perot cavity (see V&V). All the system requirements are met, and validated. The laser and photonic chip temperature are both effectively controlled using a custom TEC driver, and a sophisticated, data-driven tuning algorithm. When the user presses “run sweep” on the GUI, the laser power is transmitted through the chip and measured by photodiodes whose outputs are transformed to voltage by low-noise adjustable-gain trans-impedance amplifiers. This voltage is sampled by a low-noise ADC with configurable oversampling rate. These oversampled values are transmitted to the host computer using a custom bit packing/unpacking algorithm to reduce communication overhead and increase sweep speed. This data is then displayed to the user in real time and saved to a file for further processing.

Our system is an all-in-one photonic measurement device and is perfectly poised for members of the biosensing lab to take over and use for their experiments functionalizing photonic chips for measurement.

# 

# 8 References

[1] “A PCB about the size of a pizza box with a silicon chip in the center,” image generated by OpenAI’s DALL·E 2, (accessed Oct. 09. 2024)

[2] S. Colley, “How to Perform Transient Analysis and Noise Source Simulation with LTspice,” *Allaboutcircuits.com*, May 23, 2019. <https://www.allaboutcircuits.com/technical-articles/how-to-perform-transient-analysis-noise-simulation-LTspice/> (accessed Oct 20, 2024).

[3] Texas Instruments, OPA569 Rail-to-Rail I/O, 2A POWER AMPLIFIER Datasheet, <https://www.ti.com/lit/ds/symlink/opa191.pdf> (accessed Apr. 5, 2025).

[4]“Unlocking the Power of Aluminum PCB: Enhancing Heat Dissipation in Electronics,” *Jlcpcb.com*, 2023. <https://jlcpcb.com/blog/unlocking-the-power-of-aluminum-pcb> (accessed Oct 31, 2024)

‌[5] Lennart Ljung, *System identification : theory for the user*. Upper Saddle River, Nj: Prentice Hall Ptr, 2012.

[6] Karl Johan Astrom and Tore Hägglund, *Advanced PID control*. Research Triangle Park, N.C.: Isa, 2006.

[7] O. Garpinger, “Analysis and Design of Software-Based Optimal PID Controllers,” Doctoral thesis, Department of Automatic Control, Lund Institute of Technology, Lund University, 2015.

[8] V. R. Segovia, T. Hägglund, and K. J. Åström, “Design of Measurement Noise Filters for PID Control,” *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8359–8364, 2014, doi: https://doi.org/10.3182/20140824-6-za-1003.01486.

[9] L. F. Chaparro and A. Akan, “Introduction to the Design of Discrete Filters,” in *Signals and Systems Using MATLAB (Third edition)*, Academic Press, 2018, pp. 721–802. doi: <https://doi.org/10.1016/B978-0-12-814204-2.00023-5>.

[10] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.0 beta. <https://cvxr.com/cvx>, September 2013.

[11] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs, Recent Advances in Learning and Control (a tribute to M. Vidyasagar), V. Blondel, S. Boyd, and H. Kimura, editors, pages 95-110, Lecture Notes in Control and Information Sciences, Springer, 2008. <http://stanford.edu/~boyd/graph_dcp.html>.

[12] M. Hast, K. Åström, B. Bernhardsson, and S. Boyd, “PID Design by Convex-Concave Optimization,” Jul. 2013. Accessed: Apr. 12, 2025. [Online]. Available: <https://stanford.edu/~boyd/papers/pdf/pid_tuning_ecc.pdf>

# 

# 9 Appendix

## Appendix A: Glossary

| **Term** | **Definition** | **Context within our project** |
| --- | --- | --- |
| TEC | Thermoelectric cooler | Main actuator for photonic chip stage temperature control |
| LDC | Laser Diode Controller | Custom hardware designed and built by the client to drive a DFB laser |
| DFB | Distributed Feedback (laser) | A low-cost laser, in a butterfly package in our project |
| PID | Proportional-Integral-Derivative (control) | Control the temperature of the photonic chip |
| GUI | Graphical user interface | Software to control all aspects of our project and assist in data analysis |
| PIC | Photonic Integrated Circuit | Silicon photonic device to photonically measure biological analytes |
| DAC | Digital-to-analog convertor | Create analog control voltage for TEC controller |
| ADC | Analog-to-digital convertor | Used to capture the output voltages from the PD-TIA circuits and to measure the temperature voltage output from the thermistor analog circuit |
| TIA | Transimpedance amplifier | Used to convert the output current from the photodiodes into an output voltage |
| MCU | Microcontroller | Used to coordinate hardware functions |
| SiPh | Silicon Photonics | The overarching field behind our biosensor |

*Table A.1: Glossary of terms*

## Appendix B: PC-Firmware Interface Specification

The following tables define the packets for all transmission between the host PC and MCU.

| Commands |
| --- |
| Read Temperatures |
| Set LDC Current |
| Step LDC Current |
| Read PD Values |
| Read LDC PID Gains |
| Set LDC PID Gains |
| Read photonic chip PID Gains |
| Set photonic chip PID Gains |
| Enable LDC Driver |
| Disable LDC Driver |
| Enable LDC TEC Driver |
| Enable photonic chip TEC Driver |
| Disable photonic chip TEC Driver |

*Table A.2: Available functions to host PC*

| Field Name | Field Type | Description |
| --- | --- | --- |
| Command | uint8\_t | A one-byte value to indicate the command |
| LDC DAC Count | uint16\_t | 12-bit value to set LDC DAC |
| Kp | float | Gain for proportional term |
| Ki | float | Gain for integral term |
| Kd | float | Gain for derivative term |
| CRC | uint8\_t | 8-bit cyclic redundancy check |

*Table A.3: Packet structure for all commands from host PC*

| Field Name | Field Type | Description |
| --- | --- | --- |
| Command | uint8\_t | Issued command responding to |
| Status | uint8\_t | Status bits of eval PCB state |
| CRC | uint8\_t | 8-bit cyclic redundancy check |

*Table A.4: Packet structure for response to: Set LDC Current, Step LDC Current, Set LDC PID Gains, Set photonic chip PID Gains, Enable LDC Driver, Disable LDC Driver, Enable LDC TEC Driver, Disable LDC TEC Driver, Enable photonic chip TEC Driver, and Disable photonic chip TEC Driver*

| Field Name | Field Type | Description |
| --- | --- | --- |
| Command | uint8\_t | Issued command responding to |
| LDC ADC Count | uint16\_t | ADC Count for LDC thermistor |
| Photonic chip ADC Count | uint16\_t | ADC Count for photonic chip thermistor |
| Status | uint8\_t | Status bits of eval board state |
| CRC | uint8\_t | 8-bit cyclic redundancy check |

*Table A.5: Packet structure for response to: Read Temperatures*

| Field Name | Field Type | Description |
| --- | --- | --- |
| Command | uint8\_t | Issued command responding to |
| PD\_n ADC Count (n = 0:15) | uint16\_t | ADC count for photodiodes 0 - 15 |
| LDC ADC Count | uint16\_t | ADC count for LDC thermistor |
| Photonic chip ADC Count | uint16\_t | ADC count for photonic chip thermistor |
| Status | uint8\_t | Status bits of eval board state |
| CRC | uint8\_t | 8-bit cyclic redundancy check |

*Table A.6: Packet structure for response to: Read PD Values*

| Field Name | Field Type | Description |
| --- | --- | --- |
| Command | uint8\_t | Issued command responding to |
| Kp | float | Gain for proportional term |
| Ki | float | Gain for integral term |
| Kd | float | Gain for derivative term |
| Status | uint8\_t | Status bits of eval board state |
| CRC | uint8\_t | 8-bit cyclic redundancy check |

*Table A.6: Packet structure response to: Read LDC PID Gains and Read photonic chip PID Gains*