

Oxide Line - Laser Controller

Software Functional Requirements

By **Dr Gary Twinn**

Contents

Oxide Line - Laser Controller	1
Contents	2
List of Figures	3
List of Tables	4
Overview	5
Query Messages	7
Command Messages	7
Output Messages	9
Web interface	10
Interfaces	11
TTL convertor Schematic	11
Connections	12
Raspberry Pi Operating System Installation	13
Operating system	13
Raspberry Pi Software installation	13
Run Operating System Updates	13
Install git and gh	13
Nginx installation	13
Configure git and download the valve controller code	14
Create a python venv	14
Change to the venv	14
Install the Required Python Libraries	14
Gunicorn for Python service	14
Nginx configuration	15

List of Figures

Figure 1: The UCL Oxide Nobel Gass Extraction Line.....	5
Figure 2: Web status page	10
Figure 3: Laser controller TTL convertor and interlock configuration schematic	11
Figure 4: USB connections to the Laser Controller Raspberry Pi	12

List of Tables

Table 1: Laser status values	9
Table 2: External Equipment Connections	12
Table 3: Pyrometer serial commands.....	12

Overview

The new oxide extraction line is a configurable Nobel gas extraction line used as a test environment (Figure 1).

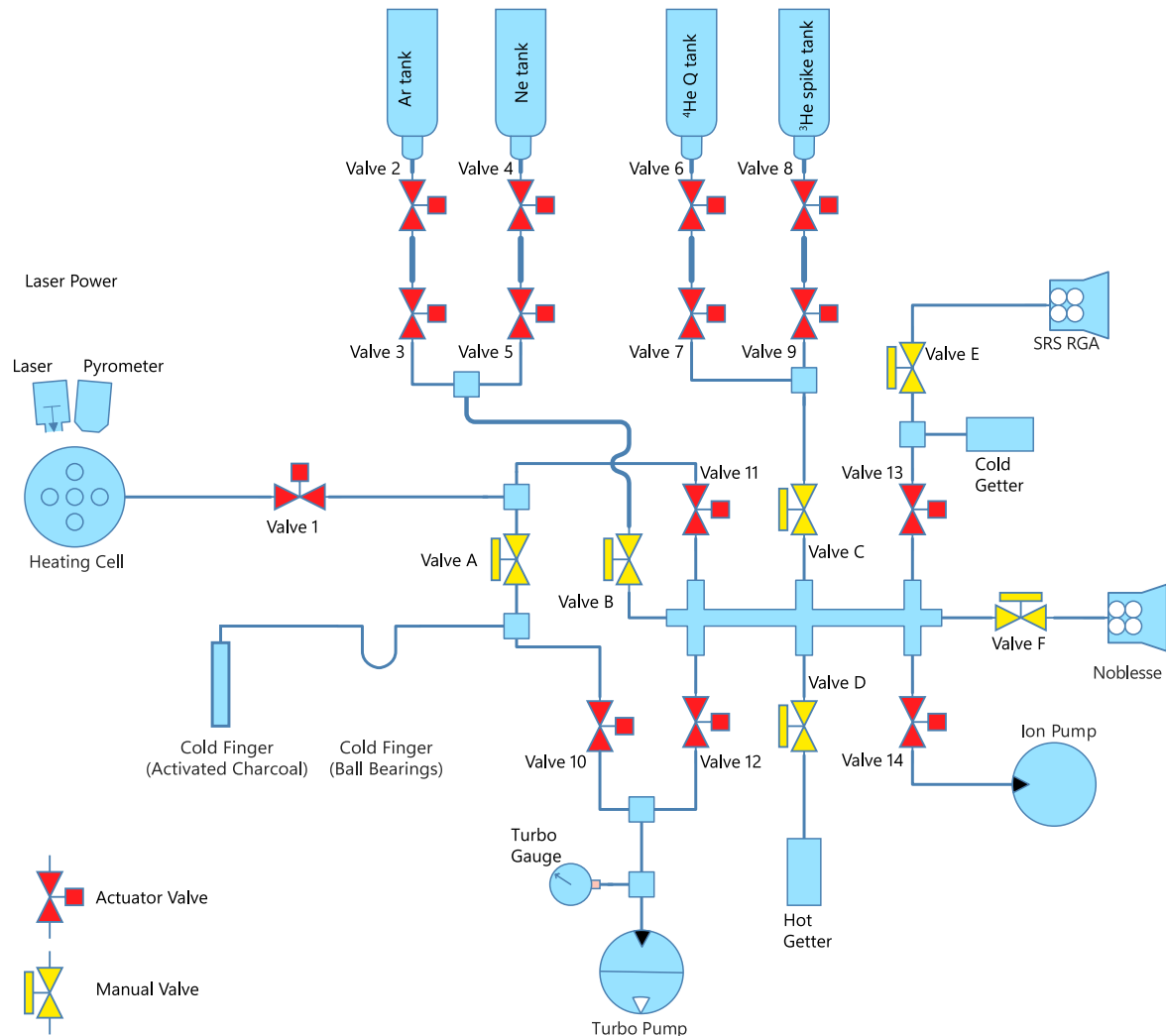


Figure 1: The UCL Oxide Nobel Gass Extraction Line

It consists of a laser/pyrometer system use to heat samples and a mass spectrometer used to measure the resultant gasses. The extraction line includes a ^3He spike tank, a ^4He standard tank as well as an Ar and Ne tank. There is a cryotrap which consists of two cold fingers that can be added into the line as well.

The laser control software includes a Python (Python Software Foundation 2023) application for managing a LaserTree K60 450nm laser (Shenzhen Guangchuangfeng Technology Co Ltd 2024).

This application uses the Flask web interface (The Pallets Project 2020) to provide an HTTP status page and handle requests via the API (Application Programming Interface).

The computer controlling the laser is a Raspberry Pi 4B (Raspberry PI Foundation 2020). The laser unit has a Transistor-Transistor-Logic (TTL) input for switching the beam on and off. The beam power can be modulated using pulse width modulation, with a duty cycle ranging from 0 to 100% to manage the output power.

To enable the beam, the laser requires a +5v TTL output, but the Raspberry Pi GPIO uses a 3.3v Complementary Metal Oxide Semiconductor (CMOS) integrated circuit. To step up the signal voltage, an SN74HC125NE 4 buffer integrated circuit (Texas Instruments 1984) was used to convert the CMOS signal to TTL for laser control.

The system includes two safety interlocks: a key switch to arm the laser and a microswitch attached to the door of the laser enclosure to prevent the laser from firing if the access door is open. Either interlock will cut the power from the laser module and inform the raspberry pi that an interlock has been triggered.

Temperature feedback is via a Micro Epsilon Infrared Pyrometer (Micro-Epsilon 2024) connected to a CH340 USB to RS232 serial interface (Nanjing Qinheng Microelectronics Co. 2019).

A Dinolite Premier USB webcam is attached to the Raspberry Pi to feed images of the laser / sample back to the PyMS application.

The computer is accessed via a RESTful API listening on port 80 for valid JSON messages. A 30-character API key is generated when the application is first run to secure the API.

Query Messages

Json messages in the following formats are accepted:

Read Laser Status:

```
{
  "item": "laserstatus",
  "command": 1
}
```

The Raspberry Pi will return the laser power level, interlock status and if the laser is firing or not.

Read Pyrometer Temperature:

```
{
  "item": "gettemperature",
  "command": "read"
}
```

Command Messages

Json messages in the following formats are accepted:

Switch on laser:

```
{
  "item": "laser",
  "command": "on"
}
```

Switch off laser:

```
{
  "item": "laser",
  "command": "off"
}
```

Set the laser power to nn.n%

```
{
  "setlaserpower": nn.n
}
```

Change the default maximum time the laser can fire to nnn seconds (default is 300)

```
{  
  "setlasertimeout": nnn  
}
```

Switch on pyrometer rangefinder laser:

```
{  
  "item": "pyrolaser",  
  "command": "on"  
}
```

Note, the pyrolaser 'on' request will also set a time to automatically switch off the laser after 60 seconds unless it is manually switched off earlier.

Switch off pyrometer rangefinder laser:

```
{  
  "item": "pyrolaser",  
  "command": "off"  
}
```


Output Messages

Following a “laserstatus” command, the following data is returned:

```
{  
  "laser":0,  
  "power":45.0,  
  "key": 0,  
  "door": 0  
}
```

The "laser" value is 1 if the laser is firing and 0 if the laser is off.

The "key" and "door" alarm values are both 1 for disabled or 0 for enabled. This allows for software to sum the two values and if the sum is zero the laser can be switched on.

ID	Value	Description
door	1	Door is open, laser disabled
door	0	Door is closed, laser can be armed if key = 0
key	1	Key is in position off, laser disabled
key	0	Key is in position on, can be armed if door = 0

Table 1: Laser status values

Following a “gettemperature” command, the following data is returned:

```
{  
  "temperature": 385,  
  "laser": "off",  
}
```

Temperature will be °C in floating point notation when connected to the pyrometer. When disconnected it will return a zero.

Web interface

As well as reading and setting the status of the laser via the RESTful API a read-only web interface that can be accessed directly from a browser and will be available to view the laser status, laser settings, temperature, webcam output, application and web server logs.

London Geochronology Centre - Oxide Line Laser Controller - Server Status CPU 37.0°C

[Return to index](#) | [Application Log](#) | [Website Access Log](#) | [Website Error Log](#) | [System Log](#)

Laser Status


Description	Status	
Laser Status	0	
Laser Power (%)	10	
Laser Auto off (s)	600	
Laser Key Switch	Key Enabled	
Laser Door Interlock	Door Closed	
Pyrometer temperature	0	
Pyrometer rangerfinder laser	0	
Pyrometer max temperature	0	
Thread	MainThread	3395
Thread	Thread-1	3397
Thread	pyro-read-thread	3398
Thread	ThreadPoolExecutor-0_0	3405
Webcam		

Figure 2: Web status page

Interfaces

TTL convertor Schematic

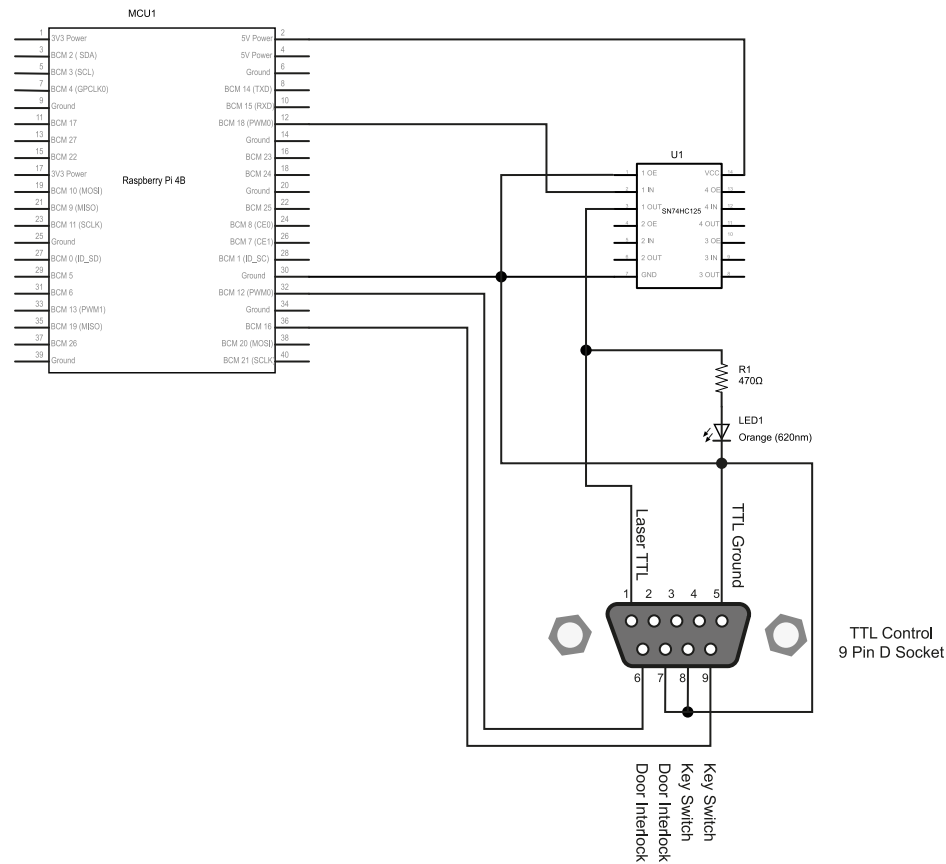


Figure 3: Laser controller TTL convertor and interlock configuration schematic

Connections

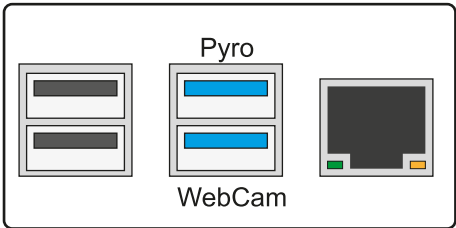


Figure 4: USB connections to the Laser Controller Raspberry Pi

Connection	USB	Linux Device	Equipment Connector
Pyrometer	USB3 port 0	/dev/ttyUSB0	RS232 Socket (female)
Webcam	USB3 port 1	/dev/video0	USB

Table 2: External Equipment Connections

Equipment	Command (Hexadecimal)
MicroEpsilon Pyrometer	01 (Read Temperature)
	05 (Read Laser Status)
	A5 01 A4 (Switch Laser On)
	A5 00 A5 (Switch Laser Off)

Table 3: Pyrometer serial commands

Raspberry Pi Operating System Installation

Operating system

Use BalenaEtcher to install the latest version of the bookworm-lite 64bit operating system onto a 32Gb MicroMMC card.

Run the `sudo raspi-config` command to:

- ♦ enable ssh
- ♦ disable Serial
- ♦ disable 1²C bus
- ♦ set the password on the pi user account to something other than “raspberry”
- ♦ change the GPU memory to 16Gb

Raspberry Pi Software installation

Run Operating System Updates

Run `sudo apt update`

Run `sudo apt upgrade`

Install git and gh

Run `sudo apt install git`

Run `sudo apt install gh`

Run `gh auth login`

- What account do you want to log into? GitHub.com
- What is your preferred protocol for Git operations? HTTPS
- Authenticate Git with your GitHub credentials? (Y/n) y
- How would you like to authenticate GitHub CLI? Paste an authentication token

Nginx installation

Run `sudo apt install nginx`

Configure git and download the valve controller code

Create a GitHub folder

Run `mkdir github`

Run `cd github`

Clone the repo

Run `git clone https://github.com/westerlymerlin/UCL-Oxide-Laser-Controller.git`

Copy the files to the home folder

Run `cp -r ~/github/UCL-Oxide-Laser-Controller/* ~/`

Set the execute flag on the scripts

Run `chmod 755 ~/bin/*`

Copy the Raspberry pi config files to the etc folder

Run `sudo cp -r ~/raspberry-pi/etc/* /etc`

Reboot the Raspberry pi

Run `sudo reboot`

After the reboot a warning banner will appear at ssh logon.

Create a python venv

Run `python3 -m venv .venv`

Change to the venv

Run `source activate`

Install the Required Python Libraries

Run `pip install -r requirements.txt`

Gunicorn for Python service

Run `sudo systemctl enable gunicorn`

Nginx configuration

Change directory to `/etc/nginx/sites-enabled/`

Run `sudo rm default`

Run `sudo ln -s /etc/nginx/sites-available/laserdata`

Reboot the Raspberry pi

Run `sudo reboot`

If flask is installed, the python files are in the `/home/pi` directory, gunicorn3 is installed and configured and nginx is installed and configured the web service should be running, and the site will be accessible on `http://ip address of the server`

References

- Micro-Epsilon (2024) Infrared Pyrometers, Available online: <https://www.micro-epsilon.com/industry-sensors/temperature-sensors/pyrometer-with-laser-sighting-ctlaser/> [Accessed 2024].
- Nanjing Qinhang Microelectronics Co., L. (2019) USB to UART Bridge Controller CH340 Datasheet, Available online: <http://wch-ic.com/products/CH340.html> [Accessed 2020].
- Python Software Foundation (2023) *Python 3 Programming Language*. Online. Available online: <https://www.python.org> [Accessed March 2023].
- Raspberry PI Foundation (2020) *Raspberry PI Model 4B Reference*, 2020. Available online: <https://www.raspberrypi.org/products/raspberry-pi-4-model-b/>.
- Shenzhen Guangchuangfeng Technology Co Ltd (2024) LaserTree K60, Available online: <https://lasertree.com/products/lasertree-k60-60w-40w-20w-power-switchable-laser-module> [Accessed 2024].
- Texas Instruments (1984) *SN74HC125NE4 Datasheet*. Available online: <https://www.ti.com/lit/ds/symlink/sn74hc125.pdf> [Accessed July 2020].
- The Pallets Project (2020) *Flask is a lightweight WSGI web application framework*. Available online: <https://palletsprojects.com/> [Accessed 2020].