Supplementary Information

Development of a Low-Cost Multi-Wavelength Inline Detector for Slug Detection in Continuous Flow

Daniel Maddox†*, Lucie Guetzoyan† and David L. Walmsley†

†Vernalis Research, Granta Park, Great Abington, Cambridge, Cambridgeshire CB21 6GB, UK

[*] Corresponding Author
Daniel Maddox
Vernalis Research,
Granta Park,
Great Abington,
Cambridge,
CB21 6GB
e-mail: d.maddox@vernalis.com

Table of Contents

S1 - General Considerations

Herein we provide full details of the design, assembly, and use of our inline detector.

The design consists of two major components: the detector module which houses the detector, fluidic piping (acting as the flow cell), a white LED and a RJ45 connector; and a Raspberry Pi 4 Model B which comprises of a standard Raspberry Pi as well as a custom HAT (Hardware Attached on Top) which contains the control electronics for the LED in addition to a RJ45 connector. The two devices are interconnected using an RJ45 cable. Both devices are housed in custom designed 3D-printed cases. A Raspberry Pi Zero should also be compatible, however the smaller 3D printed case has not been designed, nor has the lesser computational performance been tested.

The detector is operated using the Raspberry Pi, interfacing *via* a connected screen or remote desktop protocol. The supplied code plots a graph of datapoints from the detector, refreshing upon each new datapoint acquisition. The code also stores the data as a .csv file for review later, along with a .png image of the graph.

All associated files can be found on the Vernalis GitHub: https://github.com/vernalis/digital-lab/tree/main/Inline%20Detector

S2 - Bill of Materials

The following components and materials are required to construct our design:

Item	Catalogue Number	Supplier	Quantity	Total Price (£)
Raspberry Pi 4 Model B	SC0192	Thepihut.com	1	40.00
32 Gb MicroSD Card	SD-32GB	Thepihut.com	1	8.00
Adafruit AS7341 10-Channel Light / Color Sensor Breakout (STEMMA QT / Qwiic)	ADA4698	Thepihut.com	1	15.60
5mm White LED (forward voltage 3.4V, forward current 30mA)	Components within ELEGOO Upgraded Electronics Fun Kit ASIN: B01LZRV539	Amazon.co.uk	1	16.99
NPN Transistor (PN2222)			1	
10 kΩ Resistor			2	
1 kΩ Resistor			1	
Sparkfun RJ45 8-Pin Connector	PRT-00643	Pimoroni.com	2	3.00
Sparkfun RJ45 Breakout	BOB-00716	Pimoroni.com	2	2.40
2x20 pin Female GPIO Header for Raspberry Pi	COM0001	Pimoroni.com	1	1.50
Custom PCB for detector module (price for minimum order of 3 boards shown)	HRKGEPJB (https://aisler.net/p/HRKGEPJB)	Aisler.net	1	6.17
Custom PCB for Raspberry Pi HAT (price for minimum order of 3 boards shown)	QVZTMZVR (https://aisler.net/p/QVZTMZVR)	Aisler.net	1	7.55
PLA Filament for detector module	832-0264	uk.rs-online.com/web	2.7 p/g (prorated price), 37 g used	1.00
PP Filament for Raspberry Pi case feet	174-0057	uk.rs-online.com/web	5.6 p/g (prorated price), 1 g used	0.06

PETG Filament for Raspberry Pi case	190-1962	uk.rs-online.com/web	3.5 p/g (prorated price), 53 g used	1.86
Low profile (flat) LAN cable, 1m	Widely available, e.g. ASIN: B0B6WC6MRB	Amazon.co.uk	1	2.00
1/16th inch o/d PTFE tubing, with ¼-28" connection at either end	Various suppliers, approx. 15cm required, cost negligible			
			Combined total	**£106.13**

Table S1 - Bill of materials

S3 - Circuit Design

The device consists of two circuits (sensor board and LED), both are powered and controlled by the Raspberry Pi.

The sensor board is powered by 3.3V and communicates with the Pi over I^2C (thus using the SDA and SCL pins). The sensor board is therefore connected to GPIO pins 2 (I^2C SDA), 3(I^2C SCL), a 3v3 and a ground pin.

The high-power white LED has a forward voltage greater than that provided by a standard GPIO pin (3.4V versus 3.3V). As a result, the LED must be powered by a 5V pin (with 1 kΩ resistor) and then controlled by a separate GPIO pin using an NPN transistor (*via* a 20 kΩ resistor) to complete the circuit. [1] To allow dimming of the LED brightness, the controlling GPIO pin uses PWM (Pulse Width Modulation) [2], however in our initial studies we have found 100% brightness (i.e. no PWM) gave the best signal/noise result (see **S7.1**). The LED circuit is thus connected to GPIO pin 13 (PWM), a 5V and a ground pin.

In the prototype circuit design all circuits were directly connected between the component and the Raspberry Pi. In the finalised design, to allow easy and customisable length of separation between the Raspberry Pi and the detector module, as well as consolidation of 6 wires into one cable, an RJ45 ethernet cable was used. Components are attached to each other *via* the PCB circuitry, rather than the wires depicted in the original prototype diagram.

For ease of understanding both the prototype and finalised circuits are provided in the following sections, however both are essentially identical.

S3.1 - Prototype Circuit Design

Below are two schematic diagrams of the original prototype circuit, using a breadboard to facilitate easy prototyping and modification of the circuit if required.

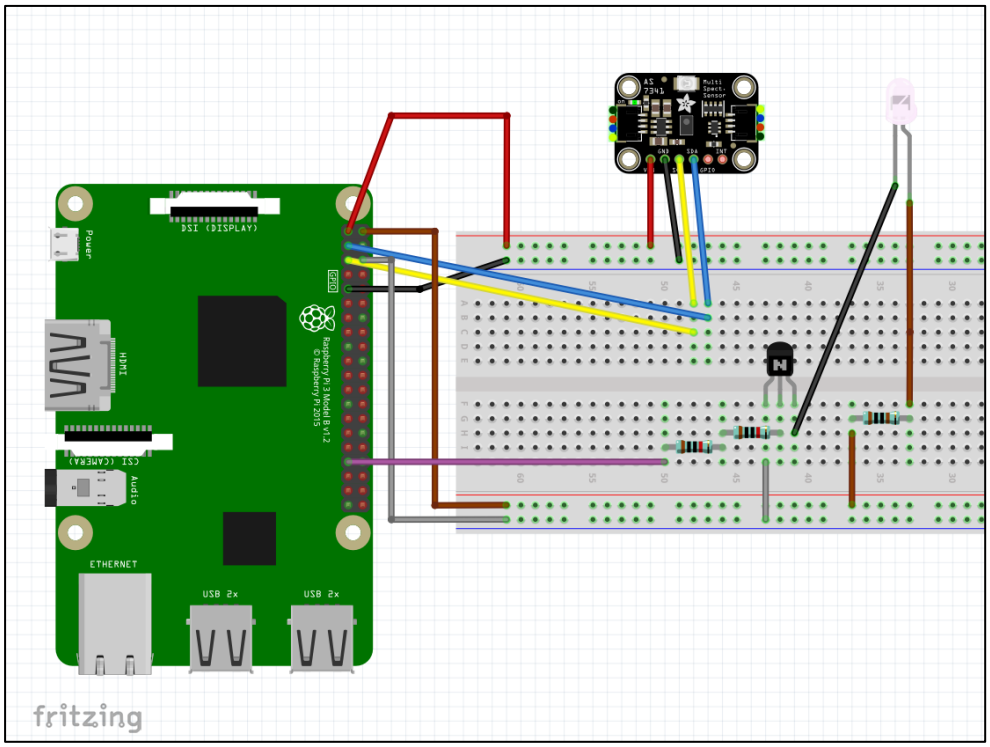

Figure S1 - Cartoon schematic of the prototype circuit

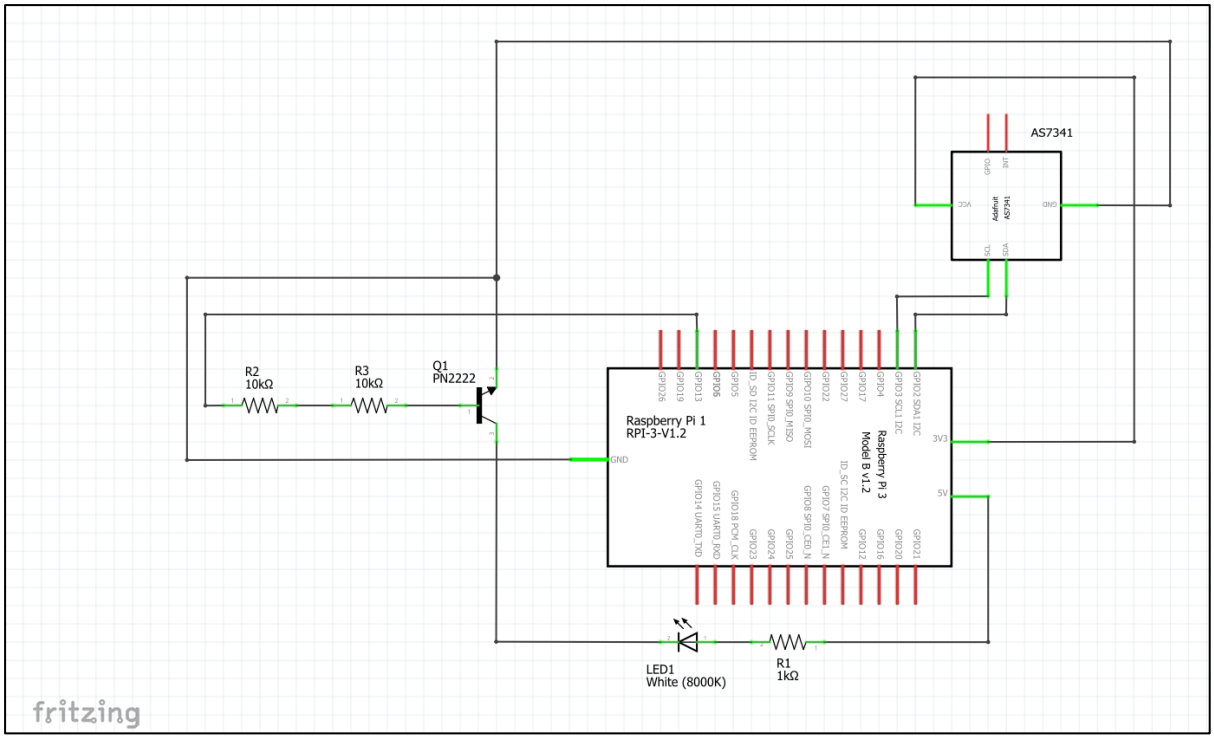

Figure S2 - Formal schematic diagram of the prototype circuit

S3.2 - Finalised Circuit Design

Below are two schematic diagrams of the finalised circuit. The cartoon representation uses a breadboard in place of the HAT circuit board.

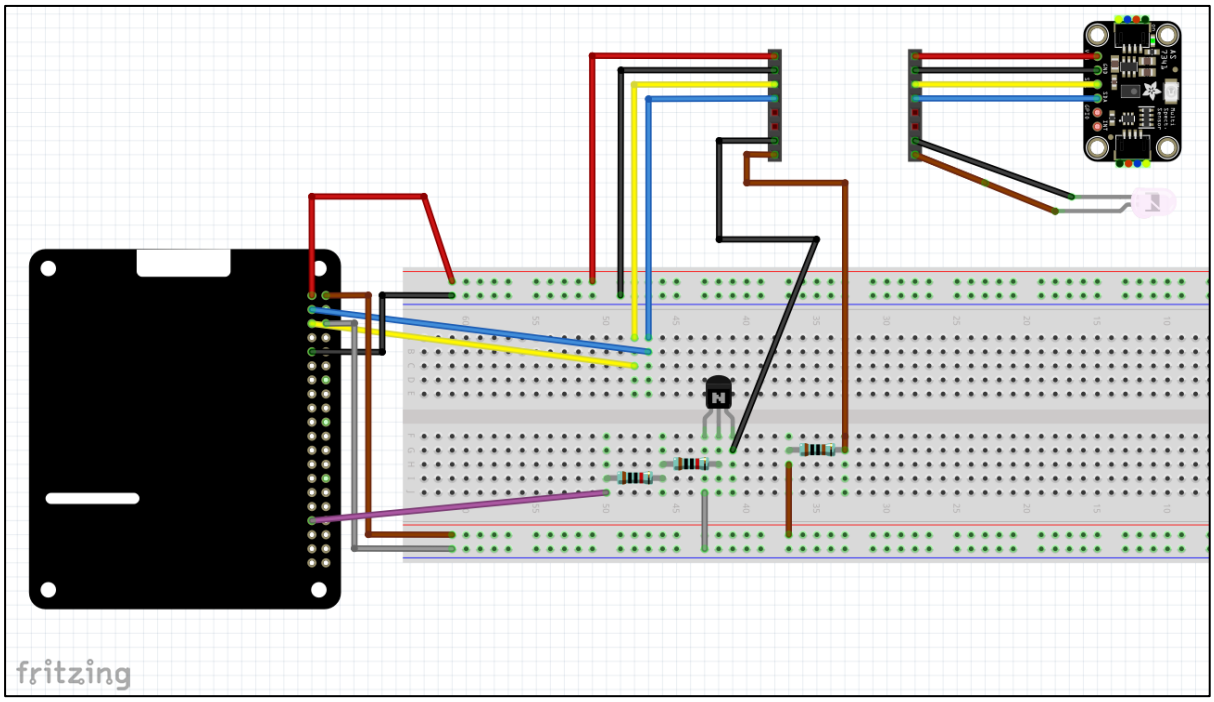

Figure S3 - Cartoon schematic of the finalised circuit

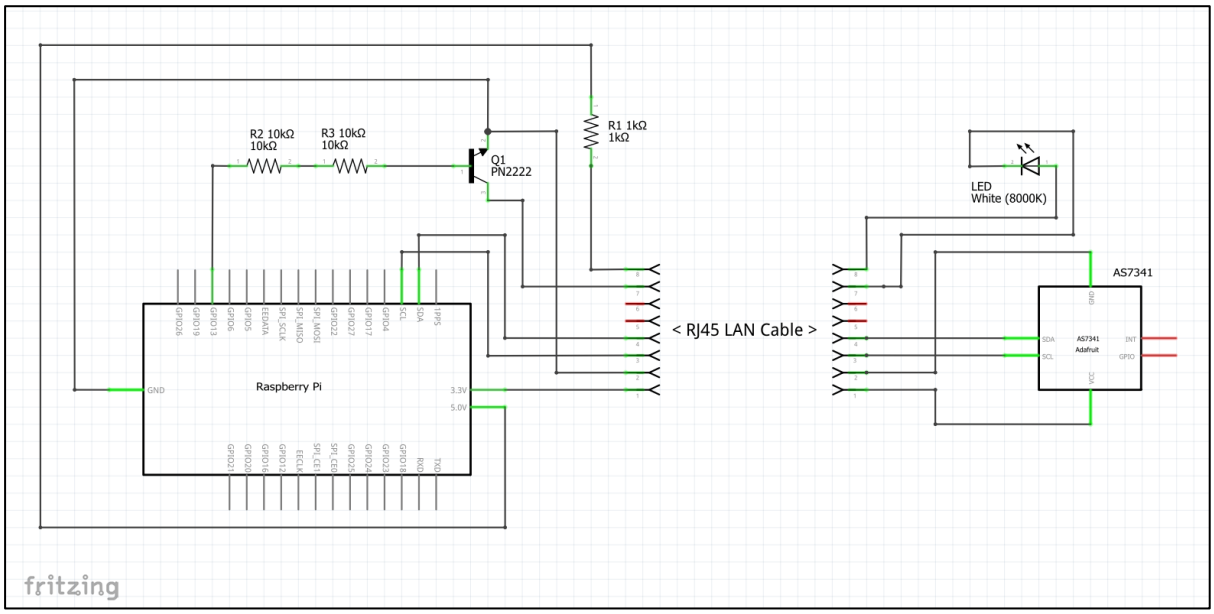

Figure S4 - Formal schematic diagram of the finalised circuit

Below are the two custom PCB designs. To keep the detector module as small as possible the sensor and LED are on one side of the board, with the RJ45 connector mounted on the underside. A female GPIO socket connector was used to enable easy attachment/detachment of the HAT to the Pi. Note: the two different colours of orange depict the two copper layers within the board. Black markings indicate silkscreen labels on the boards. The HAT is small enough to also fit a Pi Zero.

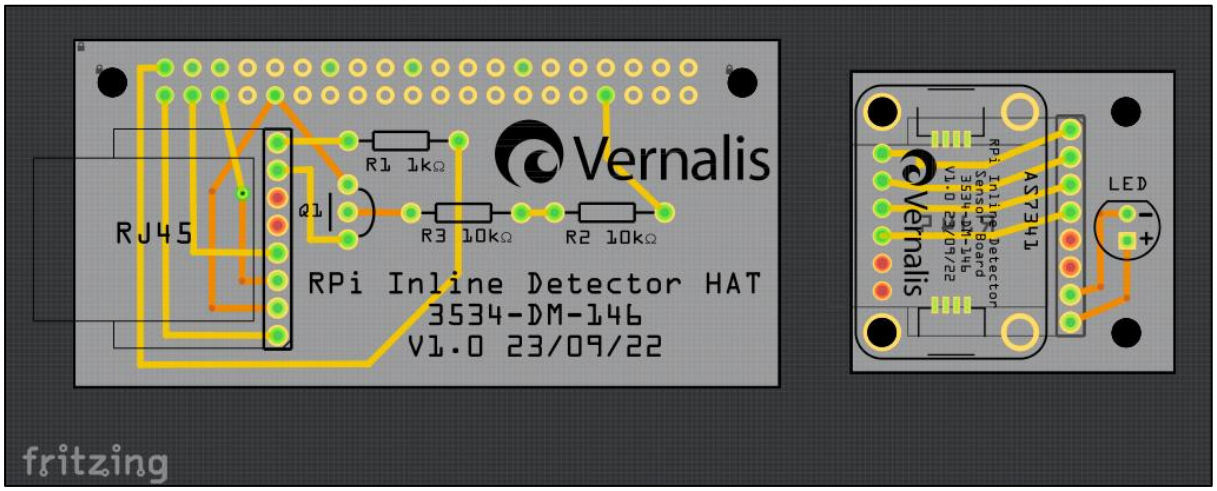

Figure S5 - Custom PCB designs

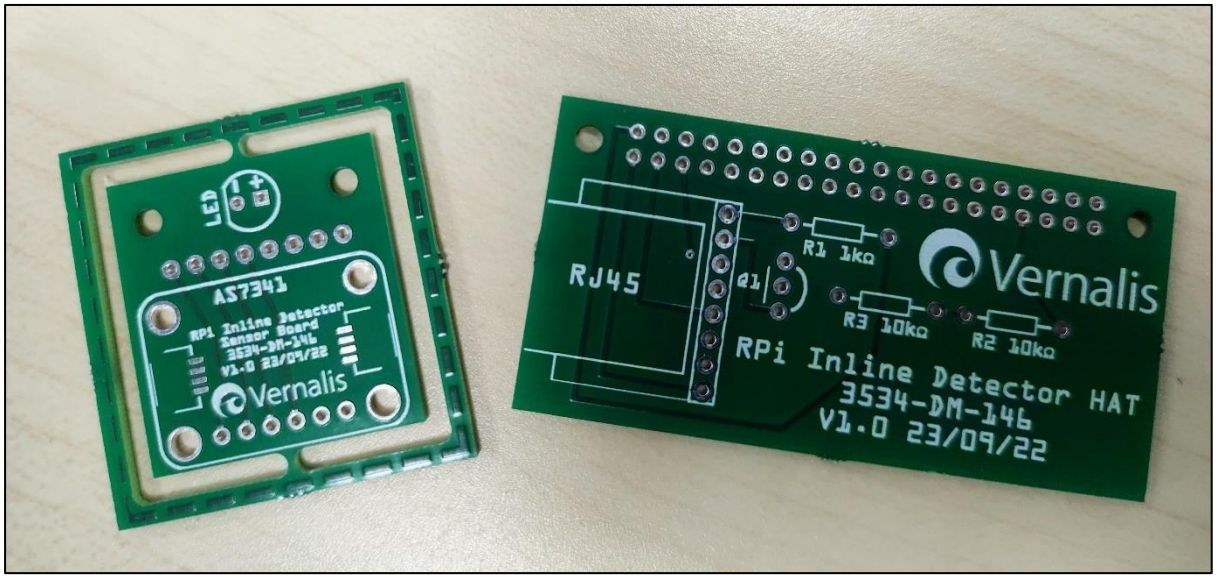

Figure S6 - Custom PCBs as delivered. Note - smaller detector board has manufacturing excess that requires removal

S3.3 - Custom Circuit Board Availability

The two custom PCB designs are available to order directly from the manufacturer we used (Aisler). To order the HAT visit https://aisler.net/p/QVZTMZVR , to order the detector module board visit https://aisler.net/p/HRKGEPJB . After registration the designs can then be imported to your account and ordered. A minimum order quantity of 3 is required, however the price is minimal (see **Table 1**).

Alternatively, the PCB Gerber files have been uploaded to our GitHub page, allowing manufacture elsewhere - https://github.com/vernalis/digital-lab/tree/main/Inline%20Detector/PCB%20Gerber%20Files

Note: The HAT PCB available at publication has undergone a minor revision to the board shown above. This revision decreases the board size by ~1.5mm, allowing compatibility with the Raspberry Pi Zero platform.

See section **S5** for assembly instructions.

S4 - Design of 3D Printed Housings

All 3D design was carried out using Tinkercad – a free, web-based CAD tool.

S4.1 - Detector Module Design

The starting point for the detector module design was the previously published "Version B" sensor housing design by Höving et al.. [3] Using the stl for this design and creating a CAD version of the Adafruit AS7341 sensor we were able to design the additional structure required to create a secure, lightproof housing for the larger sensor.

As an initial prototype, we designed a housing to simply encase the sensor (**Figure S7**).

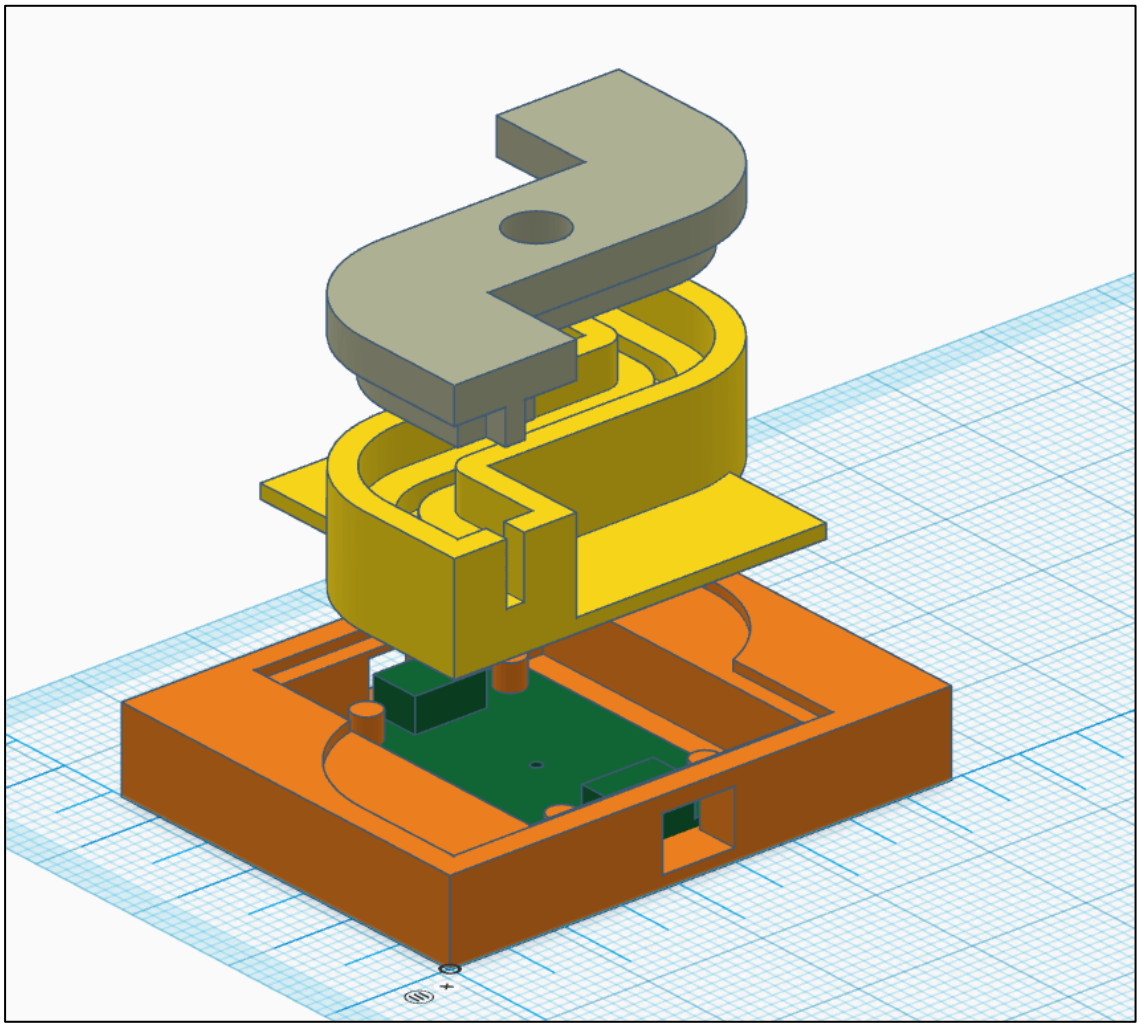

Figure S7 - Exploded CAD view of prototype detector module

Once initial success was confirmed and we moved to the finalised circuit, with custom circuit board and RJ45 link between the Raspberry Pi and detector module we designed a larger detector module housing incorporating the RJ45 connector and an access hole through which the LED wires could be fed (**Figure S8, Figure S9**).

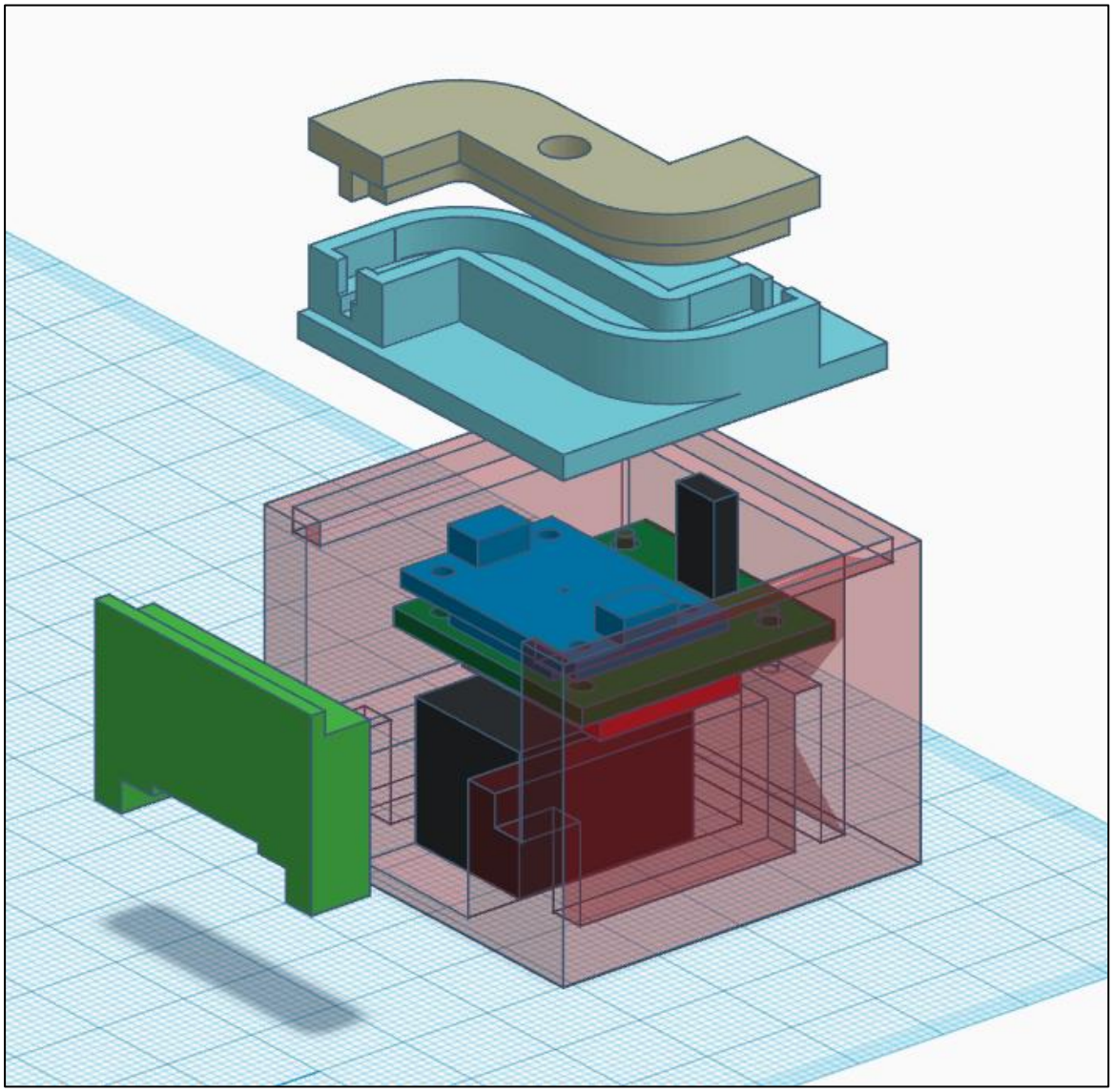

Figure S8 - Exploded CAD view of finalised detector module

Figure S9 - 3D printed and fully constructed detector module

S4.2 - Raspberry Pi Case Design

The starting point for the Raspberry Pi case design was the snap-fit design published by Malolo on the Thingiverse. [4]

The inclusion of the RJ45 connector on the custom HAT board resulted in a vertical height greater than is standard for Raspberry Pi devices. As a result, a small cut-out was made in the top surface of the case to allow the connector to stand proud. A small additional cap was then made to insert into the slot created – ensuring the RJ45 port was still protected.

In addition to this modification, to the top surface of the Pi case (and cap) was added a stippled pattern. This was for aesthetic reasons, to overcome the inconsistent patternation caused by the print bed (the case and cap are printed upside down relative to their intended orientation during use).

A further minor addition to the Malolo design was the inclusion of 4 feet to raise the bottom of the case off the fume hood floor – improving airflow, processor heat dissipation and protection from spilt solvent.

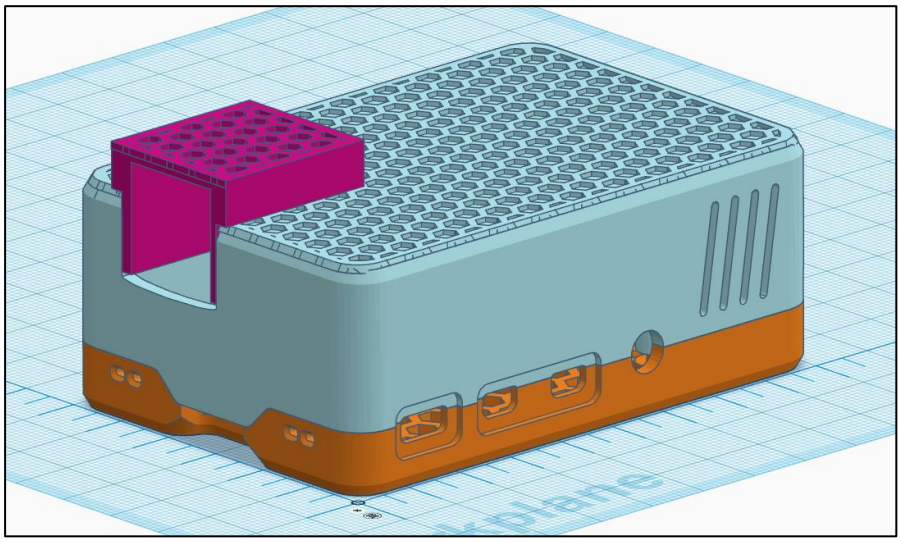

Figure S10 - Pi case CAD view: the base in orange, top in blue and RJ45 cap in pink. Feet are not shown.

Figure S11 - Constructed Pi case

S4.3 - STL File Availability

All .stl files for the finalised design are available from our GitHub page - https://github.com/vernalis/digital-lab/tree/main/Inline%20Detector/3D%20Printed%20Parts

See **S4.4** for optimal printing settings and **S5.2** & **S5.4** for assembly instructions.

S4.4 - 3D Printing Settings

All components were 3D printed using an Ultimaker S5, using 2.85mm filament.

The Raspberry Pi case was printed in polyethylene terephthalate glycol (PETG) due its higher tolerance to temperature – preventing any warping during use due to heat coming from the Pi's processor.

The Pi case feet were printed in polypropylene (PP) due to its higher solvent resistance.

The detector module had no material requirements (other than being opaque), therefore polylactic acid (PLA) was used.

When printing interlocking parts (e.g. the fluidic pipe guide) these should be printed in the same orientation as each other on the print bed to minimise any printer tolerance issues in the x and y planes.

	Material		
Condition	**Polypropylene**	**PLA**	**PETG**
Settings			
Print core temperature (°C)	240	200	245
Build plate temperature (°C)	85	60	85
Print speed (mm/s)	25	70	60
Fan speed (%)	20	20	20
Print quality			
Layer height (mm)	0.1	0.15	0.15
Wall thickness (mm)	1.14	1	1.3
Top/bottom thickness (mm)	1	1	1.2
Infill density (%)	100	20	20
Infill pattern	Octet	Triangles	Triangles
Build plate adhesion			
Prime blob	Enabled	Enabled	Enabled
Adhesion type	Brim	Brim	Brim
Plate adhesion sheet	PP	None	None

Table S2 - 3D printer settings

S5 - Assembly of Devices

S5.1 - Assembly of Detector Module Circuit Board

1. Remove excess perimeter from the supplied detector circuit board (see **Figure S6**). File off any remaining burrs.
2. Solder supplied header pins to Adafruit AS7341 breakout board. (Solder from the top face of the board, black plastic, and majority of the pin, therefore below the board).
3. Fit the AS7341 board to the top face of the detector board (silkscreen markings allow easy identification of component location) with the pins going through the holes provided. Solder the sensor in place, soldering from the bottom face of the detector board.
4. Using two header pins, solder a positive (e.g. red) and negative (e.g. black) wire to power the LED from the top face of the detector board. (**Figure S12**)

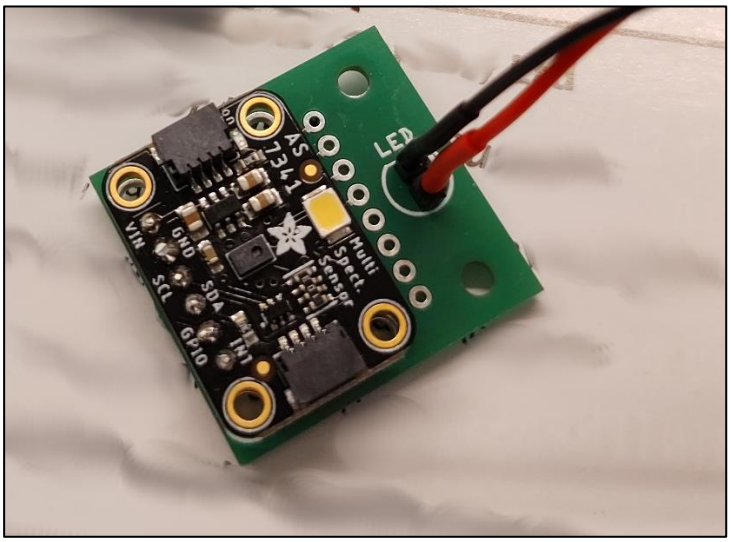

Figure S12 - AS7341 and LED wires connected to top face of detector board

5. Secure the soldered wire connections with heat shrink wrap.
6. Encase the two LED wires in a further heat shrink wrap (to create one thicker 'wire'). NOTE: do not attach the LED, or shrink the outer wrap with heat yet!
7. Push the two RJ45 parts together (the connector and the breakout board) – the two locating pegs should click into position.
8. Solder the 8 connector pins to the breakout board.

9. Attach and solder 8 header pins to the underside of the RJ45 breakout board. (**Figure S13**)

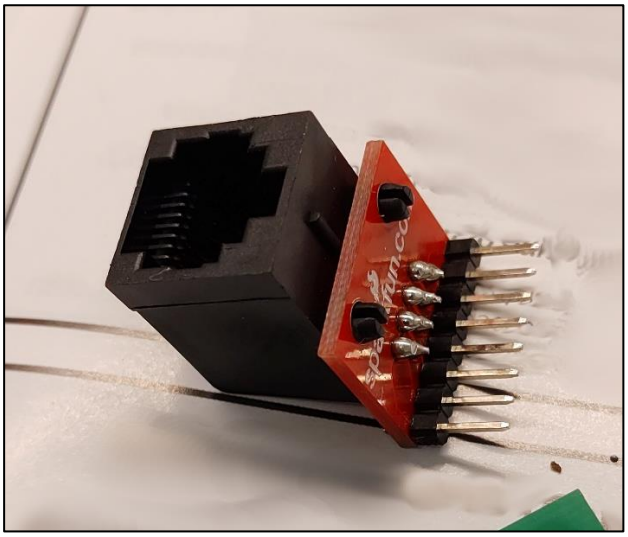

Figure S13 - Completed RJ45 breakout board

10. Fit the RJ45 breakout board to the underside of the detector board and solder in place from the top face. Detector module circuit board is now complete. (**Figure S14**)

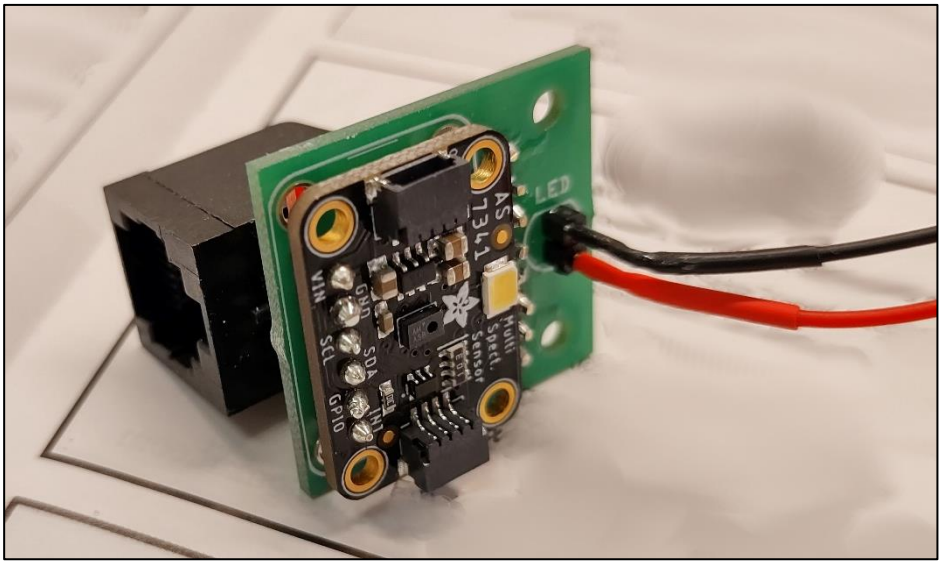

Figure S14 - Completed detector module circuit board

S5.2 - Assembly of Detector Module

1. Remove brim and any printing excess from the 3D printed parts. (**Figure S15**)

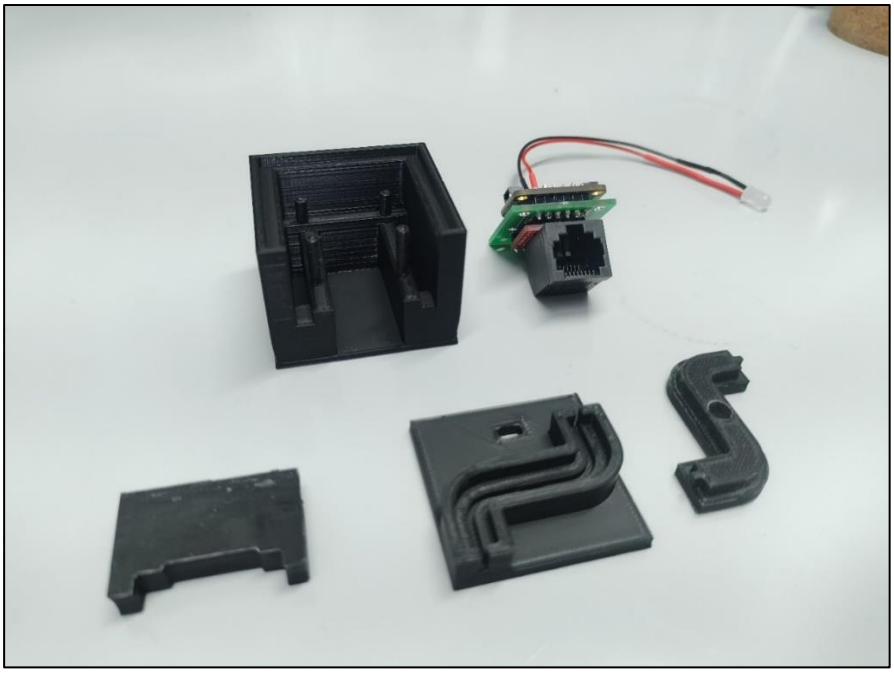

Figure S15 - Detector module components ready for assembly. Note - LED soldered in place due to prototype testing. Should not be attached at this stage.

2. Seat circuit board on the locating pegs of the main housing. Minor filing may be required due to printer tolerances. (**Figure S16**)

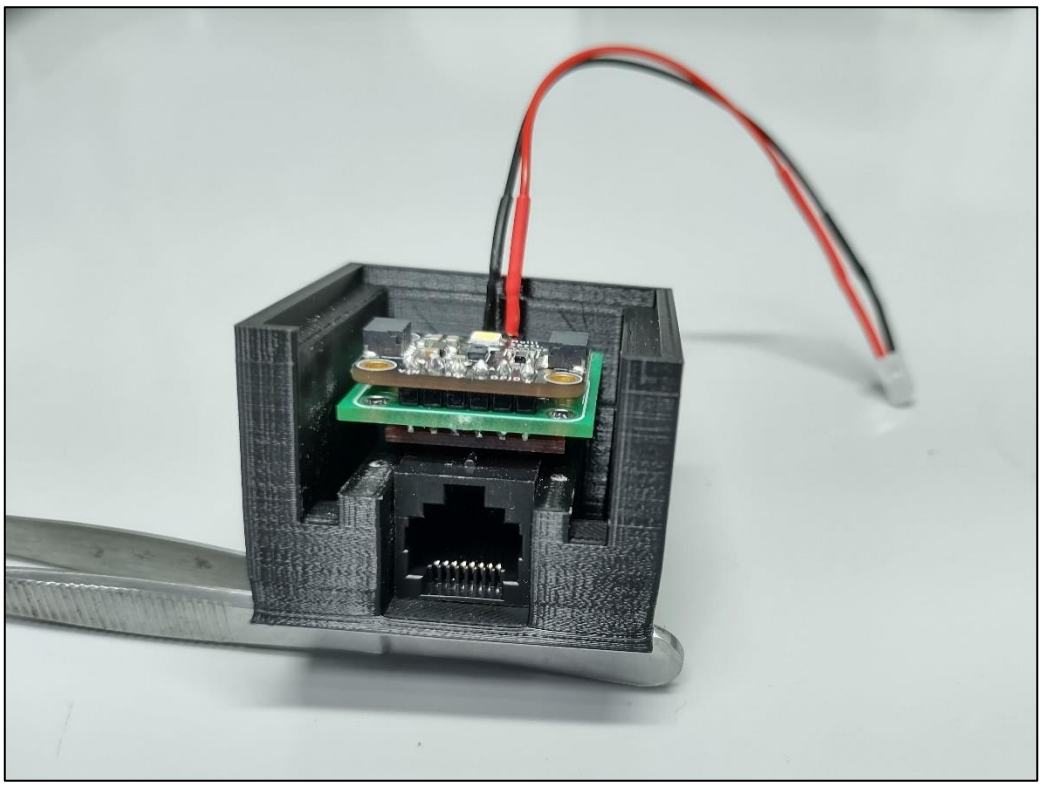

Figure S16 - Circuit board within main detector module housing

3. Fit the RJ45 cover to the wall of the detector module left open in **Figure S16**.
4. Fit a short length (approx. 15cm) of 1/16th od PTFE tubing into the pipe guide. Minor filing is likely required due to the tight tolerance (particularly at the bends). Use a smooth tool to push the tubing firmly into the channel without damaging the pipe's surface. Fit two ¼-28" fluidic connections to the pipe, allowing easy connection to a fluidic system.
5. Thread the two LED wires through the hole included in the pipe guide.
6. Solder a white LED to the other ends of the LED wires. The longer leg is the anode, therefore should be attached to the positive (red) wire. Again, protect the soldered joints with heat shrink wrap. Then heat the outer heat shrink layer, creating one single wire.
7. Push the LED into the LED top plate from the top side. This should be a tight fit, but may need minor filing depending on printer tolerances. Detector module is now complete. (**Figure S17**)

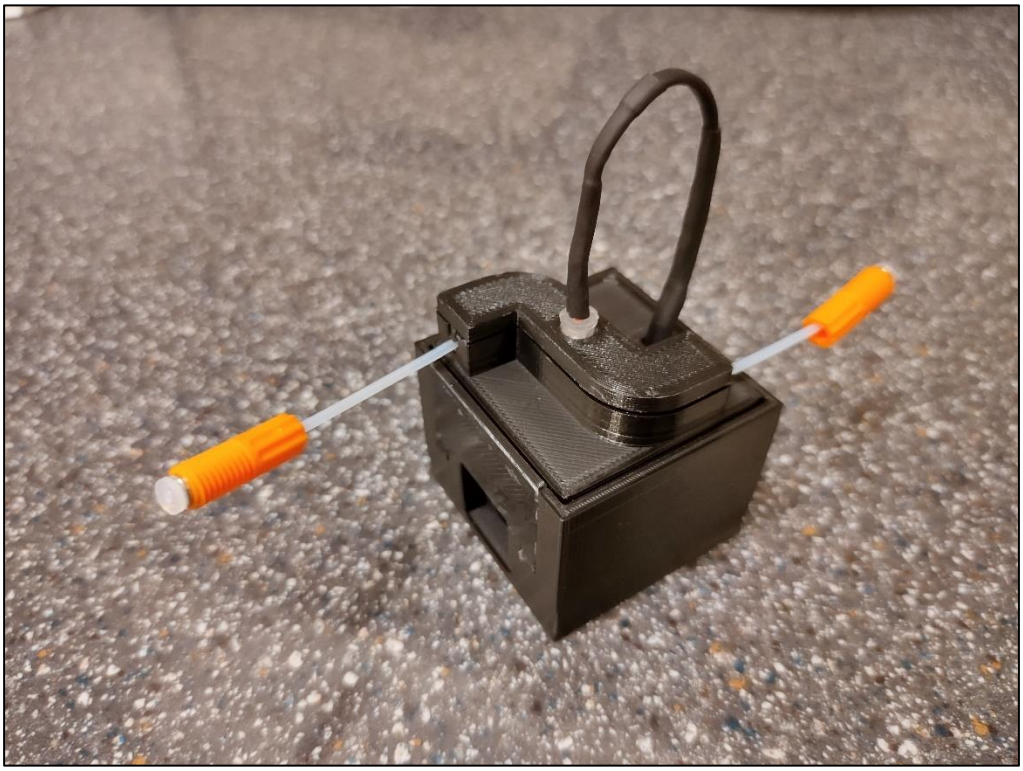

Figure S17 - Completed detector module

S5.3 - Assembly of HAT Circuit Board

1. Fit a female GPIO socket header to the underside of the HAT circuit board. Secure in place by soldering from the top face of the HAT.
2. Fit two 10 kΩ and one 1 kΩ resistors to the top face of the HAT using the indicated holes. Solder in place from the underside.
3. Fit the NPN transistor (PN2222) to the top face of the HAT using the indicated holes. Note the curved face matches the silkscreen markings on the board (i.e. flat face is nearest the RJ45 connector). Solder in place from the underside.
4. Construct the second RJ45 breakout board, following steps 7-9 from the detector module circuit board assembly instructions (**S5.1**).
5. Fit the RJ45 breakout board to the top face of the HAT using the indicated holes. Solder in place from the underside. HAT circuit board is now complete. (**Figure S18**)

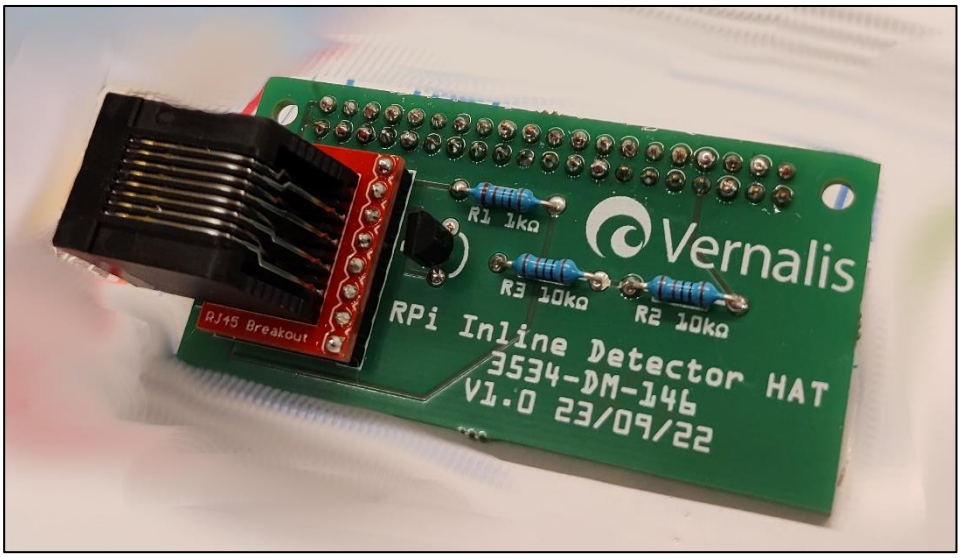

Figure S18 - Completed HAT board

S5.4 - Assembly of Raspberry Pi Module

1. Insert 32Gb SD card into Raspberry Pi
2. Attach HAT board to the GPIO pins using the female socket soldered to the bottom side of the board.
3. Remove brim and any printing excess from the 3D printed parts.
4. Attach the four poly propylene feet to the underside of the Pi case using UHU ENDFEST 300 epoxy resin. [5]
5. Carefully fit Raspberry Pi to the base part of the Pi case. Some minor filing may be required due to printer tolerances, particularly around the USB, HDMI ports on the side of the case.
6. Round off with a file the corner of the HAT between the GPIO header and the additional RJ45 port (the outermost corner of the HAT board). This is to remove any clash with the tight, curved corner of the upper part of the case.
7. Slide the top part of the Pi case into position, carefully fitting the RJ45 port on the HAT through the opening provided. The upper and lower parts of the case should click together.
8. Fit the RJ45 cap to the top part of the case to protect the RJ45 port. Minor filing may be required to accommodate printer and assembly tolerances.
9. Use a LAN cable to connect the Pi to the detector module. Assembly is complete. (**Figure S19**)

Figure S19 - Completely assembled inline detector

S6 - Raspberry Pi Configuration and Operation

S6.1 - Configuring Raspberry Pi for Use

The 32Gb SD card inserted in the Raspberry Pi contained a standard Raspberry Pi OS (bullseye) image installed.

A WiFi connection was configured, followed by enabling VNC and I^2C protocols. VNC remote desktop was used to remove the requirement for a screen, keyboard, and mouse to be connected to the device. The sensor chip communicates over I^2C.

The requisite python 3 libraries for the project were then installed from the command terminal in the following order:

1. Adafruit CircuitPython (Blinka) library, following the guide published by Adafruit [6]
2. Adafruit AS7341 specific library using the command:
   sudo pip3 install adafruit-circuitpython-as7341
3. Matplotlib using the following sequential commands:
   3.1. sudo apt install libgfortran5 libatlas3-base
   3.2. sudo pip3 install --upgrade numpy
   3.3. *sudo pip3 install matplotlib*
4. Pandas using the command: *sudo pip3 install pandas*

Following the successful installation of the libraries above. Transfer the source code available on our GitHub to the Raspberry Pi and execute - https://github.com/vernalis/digital-lab/tree/main/Inline%20Detector/Code

S6.2 - Inline Detector Operation

The inline detector was accessed *via* VNC to enable a graphical interface. Pumping of the system solvent was commenced before the inline detector code was initiated.

During execution of the code three initial sensor readings are taken to establish a baseline absorbance reading for the system solvent. Once the baseline has been established, new sensor readings are recorded approximately every 2.5 seconds. A graph plotting normalised absorbance values is re-rendered following each reading, with time on the x axis and relative absorbance on the y axis. Slugs can then be detected by observing of a change in absorbance (see **Figure S22**).

The raw absorbance values and the baseline-normalised absorbance values are saved in two separate .csv files for later reference if required. These files are regenerated every time a sensor reading is taken.

Upon completion of the flow experiment, the inline detector application should be terminated by closing the chart window. This will generate a .png image of the graph for later reference, switch off the LED and then terminate the code.

S6.3 - Firmware and Software Versions Used

In the unlikely event that a future Raspberry Pi or python library update introduces an incompatibility issue, below summarises the specific version numbers of the firmware and software libraries used on our device. If incompatibility is found with the latest releases installed using the steps detailed in **S6.1**, rolling back to the revision we used should resolve any unintended incompatibility.

Raspberry Pi OS release: Raspbian GNU/Linux 11 (bullseye)
Debian Version 11.4
Kernel: Linux 5.15.61-v7l+
Bootloader: Tue 16 Feb 13:23:36 UTC 2021 (1613481816)
Libgfortran5 version: libgfortran5-10.2.1-6+rpi1
Libatlas3-base version: libatlas3-base-3.10.3-10+rpi1

The following Python package versions were installed at time of publication. Downloading "requirements.txt" from our GitHub repository and using the command "*pip install -r requirements.txt*" will mimic this installation.

Package	**Version**
Adafruit-Blinka	8.2.0
adafruit-circuitpython-as7341	1.2.9
adafruit-circuitpython-busdevice	5.2.1
adafruit-circuitpython-register	1.9.11
adafruit-circuitpython-typing	1.7.2
Adafruit-PlatformDetect	3.27.0
Adafruit-PureIO	1.1.9
adafruit-python-shell	1.3.3
arandr	0.1.10
args	0.1.0
astroid	2.5.1
asttokens	2.0.4
automationhat	0.2.0
beautifulsoup4	4.9.3
blinker	1.4
blinkt	0.1.2
buttonshim	0.0.2
Cap1xxx	0.1.3
certifi	2020.6.20
chardet	4.0.0
click	7.1.2
clint	0.5.1
colorama	0.4.4
colorzero	1.1
cryptography	3.3.2

Package	**Version**
pexpect	4.8.0
pgzero	1.2
phatbeat	0.1.1
pianohat	0.1.0
picamera	1.13
piglow	1.2.5
pigpio	1.78
Pillow	8.1.2
pip	20.3.4
psutil	5.8.0
pycairo	1.16.2
pycups	2.0.1
pyftdi	0.54.0
pygame	1.9.6
Pygments	2.7.1
PyGObject	3.38.0
pyinotify	0.9.6
PyJWT	1.7.1
pylint	2.7.2
pyOpenSSL	20.0.1
pyparsing	3.0.9
pyserial	3.5b0
pysmbc	1.0.23
python-apt	2.2.1
python-dateutil	2.8.2

Package	Version
cupshelpers	1
cycler	0.11.0
dbus-python	1.2.16
distro	1.5.0
docutils	0.16
drumhat	0.1.0
envirophat	1.0.0
ExplorerHAT	0.4.2
Flask	1.1.2
fonttools	4.34.4
fourletterphat	0.1.0
gpiozero	1.6.2
html5lib	1.1
idna	2.1
isort	5.6.4
itsdangerous	1.1.0
jedi	0.18.0
Jinja2	2.11.3
kiwisolver	1.4.4
lazy-object-proxy	0.0.0
logilab-common	1.8.1
lxml	4.6.3
MarkupSafe	1.1.1
matplotlib	3.5.3
mccabe	0.6.1
microdotphat	0.2.1
mote	0.0.4
motephat	0.0.3
mypy	0.812
mypy-extensions	0.4.3
numpy	1.23.1
oauthlib	3.1.0
olefile	0.46
packaging	21.3
pandas	1.4.3
pantilthat	0.0.7
parso	0.8.1

Package	Version
pytz	2022.1
pyusb	1.2.1
rainbowhat	0.1.0
reportlab	3.5.59
requests	2.25.1
requests-oauthlib	1.0.0
responses	0.12.1
roman	2.0.0
rpi-ws281x	4.3.4
RPi.GPIO	0.7.1
RTIMULib	7.2.1
scrollphat	0.0.7
scrollphathd	1.2.1
Send2Trash	1.6.0b1
sense-hat	2.4.0
setuptools	63.4.3
simplejson	3.17.2
six	1.16.0
skywriter	0.0.7
sn3218	1.2.7
soupsieve	2.2.1
spidev	3.5
ssh-import-id	5.1
sysv-ipc	1.1.0
thonny	3.3.14
toml	0.10.1
touchphat	0.0.1
twython	3.8.2
typed-ast	1.4.2
typing-extensions	3.7.4.3
unicornhathd	0.0.4
urllib3	1.26.5
webencodings	0.5.1
Werkzeug	1.0.1
wheel	0.34.2
wrapt	1.12.1

Table S3 - Installed python packages at time of development

S7 - Testing

S7.1 - Initial Optimisation

The device was designed with the ability to control LED brightness, as a result two experiments were performed: one with LED brightness set to 50% and one at 100%. It was found 100% brightness gave the best signal to noise ratio and wasn't too bright to obscure subtle changes in absorbance.

The sensor provides raw brightness values across a range of wavelengths, as a result the inherent absorbance of the system solvent is observed as a consistent non-zero value (**Figure S20, A**). To make identification of slugs easier, sensor values can be normalised by deduction of a mean absorbance value arising due to the carrier solvent. This results in an approximately zero value when there is just system solvent passing the detector (baseline), and either a positive or negative peak visible (dependent on the differing absorbance characteristics) when a slug is passing the detector (**Figure S20, B**).

Figure S20 demonstrates how normalisation of the sensor values makes slug detection much easier.

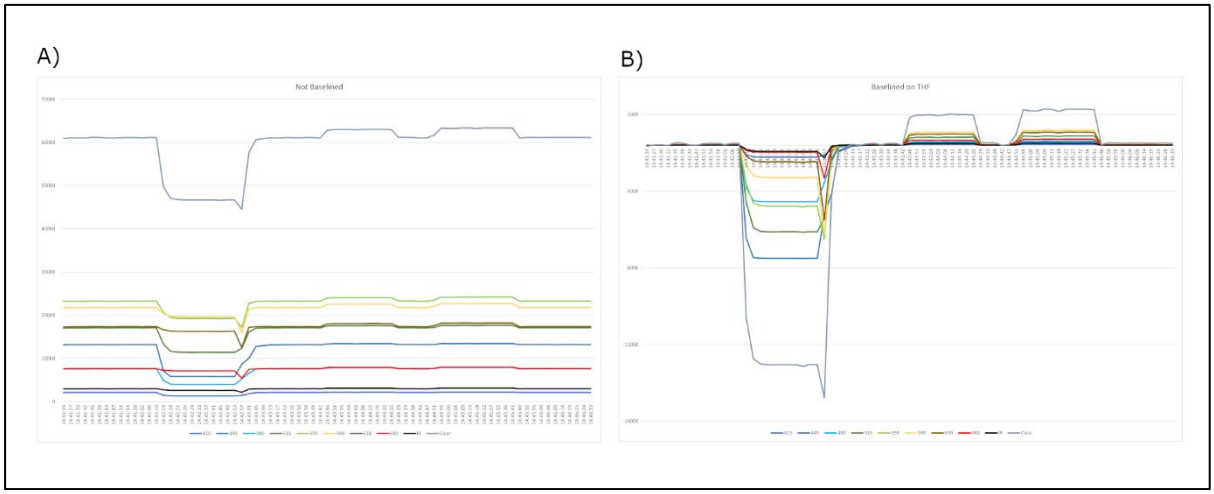

Figure S20 - Comparison of data representation without (A) and with (B) sensor value normalisation against THF system solvent baseline

The supplied code therefore operates under these optimal conditions – 100% LED brightness, with normalised values plotted on the graph.

S7.2 - Validation of Detector

Once optimal detector settings were determined and the design was finalised a validation experiment was performed by incorporating the sensor into a Syrris Asia flow chemistry system.

A simple fluidic system (**Figure S21**) was configured with both Syrris Asia AutoRIM injectors on the same single channel, this was to enable a swift experiment with rapid sequential injections. By having two injection loops on one reaction channel one injection loop could be washing and refilling whilst the other was injecting into the reactor stream. The injected slugs (2ml) then passed the detector at a flow rate of 1ml/min before exiting the fluidic system, *via* a Back Pressure Regulator, BPR, set to 2 bar.

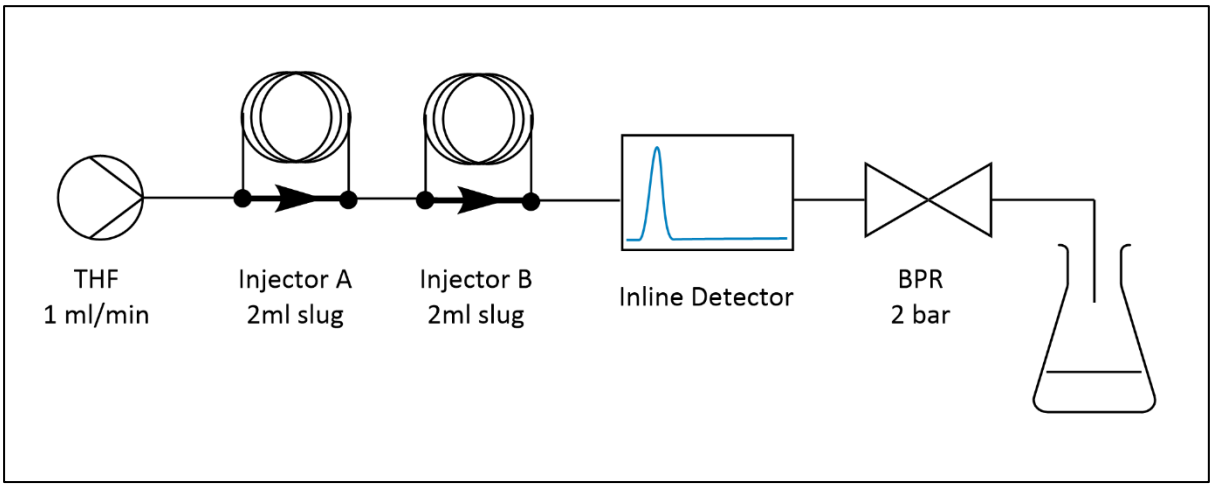

Figure S21 - Fluidic scheme for detector validation experiment

For the validation experiment THF was chosen because it is a typical solvent used within our system. Three different solutions were then injected as slugs: a dilute solution of iodine in THF, this coloured solution should be easily detectable; neat IPA (isopropyl alcohol) and finally neat ethyl acetate these were both chosen as they are colourless liquids and would be difficult to discern from THF by eye.

The resulting data output from the validation experiment is shown in **Figure S22**. All three slugs can be seen clearly, indicating that the detector works well – even for colourless solutions in a colourless carrier solvent stream.

The sharp negative spike before the iodine and IPA slugs is due to the Syrris Asia AutoRIM. To ensure a sharp slug/system solvent interface a small nitrogen bubble at the system pressure is introduced at either end of the slug. The nitrogen bubble has its own unique absorbance profile. It is not clear why the bubble wasn't observed for the ethyl acetate slug. A bubble was not introduced at the back of the slug in this experiment because slug timing was operated by manual control, and the injection was halted before the nitrogen bubble had exited the injection loop.

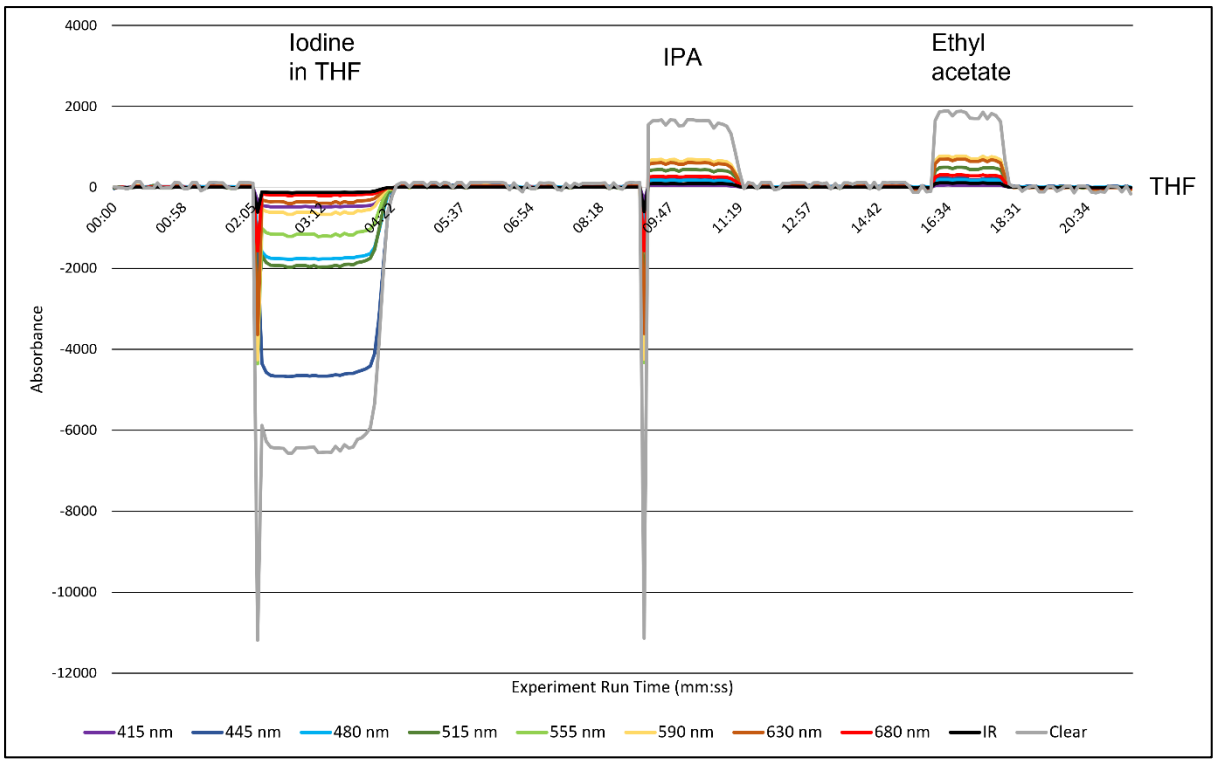

Figure S22 - Results of validation experiment. Colourless IPA and ethyl acetate slugs are clearly visible against colourless THF carrier solvent

S8 - References

[1] https://www.petervis.com/Raspberry_PI/Driving_LEDs_with_CMOS_and_TTL_Outputs/Driving_an_LED_Using_Transistors.html

[2] https://circuitdigest.com/microcontroller-projects/raspberry-pi-pwm-tutorial

[3] Höving, S., Bobers, J. & Kockmann, N. Open-source multi-purpose sensor for measurements in continuous capillary flow. *J Flow Chem* **12**, 185–196 (2022) https://doi.org/10.1007/s41981-021-00214-w

[4] https://www.thingiverse.com/thing:3723561

[5] https://www.uhu.com/en/product-page.63251

[6] https://learn.adafruit.com/circuitpython-on-raspberrypi-linux/installing-circuitpython-on-raspberry-pi