HardwareX 24 (2025) e00696

Contents lists available at ScienceDirect

HardwareX

journal homepage: www.elsevier.com/locate/ohx

Hardware article

A novel clinical data acquisition device: Towards real time cardiovascular modelling in the ICU

James Cushway [a,b,*], J. Geoffrey Chase [a], Thomas Desaive [b], Liam Murphy [a], Isaac L. Flett [a], Geoffrey M. Shaw [c]

[a] University of Canterbury, Department of Mechanical Engineering, 69 Creyke Road, Ilam, Christchurch 8041, New Zealand
[b] University of Liège (ULg), GIGA-Cardiovascular Sciences, B34, Quartier Hôpital, Avenue de l'Hôpital 11, Liège 4000, Belgium
[c] Dept of Intensive Care, Christchurch Hospital, 2 Riccarton Avenue, Christchurch Central City, Christchurch 4710, New Zealand

ARTICLE INFO

Keywords:
Customizable data acquisition system
Hemodynamic analysis
Invasive arterial pressure
Fluid infusion rate

ABSTRACT

Patient specific cardiovascular system models have the potential to provide far greater insights into patient conditions than is currently possible in the intensive care unit (ICU). Access to haemodynamic data is imperative for the introduction and validation of these models for model-based care in the ICU. However, current bedside machines are proprietary systems and do not allow access to any data without potentially prohibitive cost, creating a barrier to research and advancements in bedside care. This work presents a novel, low-cost data capture system which allows real time capture of haemodynamic measurements, as well as fluid infusion rates in the ICU. The system consists of an Arduino controlled data capture unit, an adaptor for connecting to existing bedside pressure sensors, and a Python based application which displays and records data sent from the data capture system over a USB serial connection. The total system is highly customizable if needed, and costs a total of NZ$200.

Specifications table

Hardware name	HeartView 2.0
Subject area	Biomedical engineering
Hardware type	Measuring physical properties
Closest commercial analog	The closest commercial analog equivalent is the Philips intellivue system which is currently used in most ICU's all over the world. The most recent version can be found here.
Open source license	CC BY-SA 4.0
Cost of hardware	NZ$199
Source file repository	https://doi.org/10.17605/OSF.IO/TP24R

[*] Corresponding author at: University of Canterbury, Department of Mechanical Engineering, 69 Creyke Road, Ilam, Christchurch 8041, New Zealand.
E-mail address: james.cushway@pg.canterbury.ac.nz (J. Cushway).

https://doi.org/10.1016/j.ohx.2025.e00696
Received 7 April 2025; Received in revised form 20 August 2025; Accepted 21 August 2025

Available online 11 September 2025

1. Hardware in context

Bedside monitoring is a vital part of patient care in the intensive care unit (ICU). Blood pressure monitoring has been reported as early as 1856 when Faivre connected an artery to a mercury manometer during an operation [1]. In modern bedside care, there exist 4 major vital signs: heart rate, respiratory rate, temperature and blood pressure [2]. These vital signs form a key part of decision making in the ICU, and are crucial for guiding fluid therapy.

Fluid therapy is the primary procedure used for combatting cardiovascular shock [3–5]. Fluid therapy is the administration of intravenous fluids to increase the mean systemic filling pressure, increasing the pressure differential driving venous return and thus, increasing cardiac output (CO). Currently, fluid therapy is guided either by static indices of fluid responsiveness, such as central venous pressure, pulmonary artery occlusion pressure, and global end diastolic volume, or dynamic indices, such as pulse pressure variation and stroke volume variation.

Dynamic indices have proved better than static indices [6,7]. However, large variations in the performance of dynamic indices exist throughout the medical literature [8]. Furthermore, dynamic indices are based on cardiopulmonary interactions occurring during mechanical ventilation. Thus, dynamic indices cannot be used for any patients with open thorax, or who are ventilated using spontaneous breathing [8].

Model based indicators, such as stressed blood volume (SBV), have shown promising results as an indicator for fluid responsiveness [3–5,8–10]. However, these indices are not currently available to clinicians and bedside nurses, as bedside haemodynamic data is not readily available for input into real-time cardiovascular models, due to the proprietary systems currently employed in ICUs globally.

This work presents a novel, easy to use data capture system for obtaining haemodynamic data, as well as fluid infusion rates in the ICU. This system opens up the possibility for real time cardiovascular modelling, providing much higher levels of insight into current patient condition, and thus a better basis for critical care decision making.

2. Hardware description

The hardware consists of 2 main pieces: (1) the main data acquisition device (DAQ); and (2) a custom designed adaptor enabling real time data capture from bedside machines. The DAQ consists of a custom printed circuit board (PCB), and an Arduino nano microcontroller. The PCB uses 4 RJ-45 ports as data input lines, 3 for pressure sensor inputs and 1 for a load cell input. Inputs are fed through INA821 instrumentation amplifiers before being sent to the analog to digital converter (ADC) inputs on the Arduino nano. The DAQ has a USB-C port, and the PCB is powered by the 5 V supply of the Arduino, which is connected to a computer via USB. The DAQ uses a USB-C The data read by the ADCs is sent over serial connection to the computer, where data is collected and displayed using a Python based desktop application. The adaptor is used to create a T-branch between pressure sensors and the bedside machine. The adaptor keeps the connection between bedside monitor and pressure sensor intact, but also allows the pressure signal to be sent downstream to the DAQ. The schematic for the DAQ is shown in Fig. 1.

The device is based on a previously created device, a low cost circulatory pressure acquisition system designed by Smith et al. [11]. Smith's device required the addition of extra pressure sensors to hydraulically tap into existing pressure sensors. While a good design in theory, this approach necessitated an extra pressure sensor, as well as the cutting of an RJ-45 and Philips IBP cable, and soldering them together, yielding a messy solution not suitable for hospital use. Furthermore, the addition of the extra pressure sensor presents a complex solution for bedside nurses. The adaptor presented here bypasses all of these issues, making a very easy to use system for bedside nurses, and removing the need for extra pressure sensors or chopped and re-soldered cables.

2.1. Fault tolerance and electrical isolation

Since the hardware device is connected to pressure sensors, which are connected to patients, precautions must be taken to ensure that patients are not at risk of potential electrocution. Large resistors (150k Ω) have been added on the pressure sensor input lines on the PCB. In the highly improbable fault case where VCC (regulated 5V) is applied outward from the differential amplifiers towards the patient, the resistors limit the possible reverse current flowing out of the board towards the pressure sensor to 33μA, ignoring the high impedance of the amplifiers themselves, and the additional high resistance incurred downstream of the pressure sensor. This meets the Standards New Zealand requirement for maximum allowed leakage current of 50μA outlined in the standards prescribed for medical electrical equipment in IEC 60601.1:2015.

Standard practice for medical devices is to use opto-isolators, and the future iterations of the board would benefit from this addition.

Design files

3. Design files summary

3.1. CAD files:

Most CAD files, with the exception of the hanging scale base and lid, have been drawn using Solidworks. The housing for the hanging scale, specifically the hanging scale base and lid, were created by Smith et al. [11] using FreeCAD, and were unaltered by this work. The rest of the housings, the DAQ housing base and lid, adaptor housing base and lid, as well as the modelled male

Table 1
Design file summary with locations in online repository.

Design filename	File type	Open source license	File Location
Arduino_read.ino	Code for Arduino	CC BY-SA 4.0	Software/
GUI_main.py	Python file	CC BY-SA 4.0	Software/
GUI_setup_exe.py	Python file	CC BY-SA 4.0	Software/
Loadcell_Housing_Base.FCStd	FreeCAD file	CC BY-SA 4.0	Hanging Scale/
Loadcell_Housing_Lid.FCStd	FreeCAD file	CC BY-SA 4.0	Hanging Scale/
Loadcell_Housing_Base.stl	3D-printable file	CC BY-SA 4.0	Hanging Scale/
Loadcell_Housing_Lid.stl	3D-printable file	CC BY-SA 4.0	Hanging Scale/
DAQ_Housing_Base.stl	3D-printable file	CC BY-SA 4.0	DAQ/
DAQ_Housing_Lid.stl	3D-printable file	CC BY-SA 4.0	DAQ/
DAQ_Housing_Base.stl	3D-printable file	CC BY-SA 4.0	DAQ/
DAQ_Housing_Lid.stl	3D-printable file	CC BY-SA 4.0	DAQ/
DAQ_PCB.zip	KiCAD Project	CC BY-SA 4.0	DAQ/
Adaptor_Housing_Base.sldprt	Solidworks file	CC BY-SA 4.0	Adaptor/
Adaptor_Housing_Lid.sldprt	Solidworks file	CC BY-SA 4.0	Adaptor/
Adaptor_Housing_Base.stl	3D-printable file	CC BY-SA 4.0	Adaptor/
Adaptor_Housing_Lid.stl	3D-printable file	CC BY-SA 4.0	Adaptor/
EdwardsFemaleConnector.sldprt	Solidworks file	CC BY-SA 4.0	Adaptor/
EdwardsMaleConnector.sldprt	Solidworks file	CC BY-SA 4.0	Adaptor/
EdwardsFemaleConnector.stl	3D-printable file	CC BY-SA 4.0	Adaptor/
EdwardsMaleConnector.stl	3D-printable file	CC BY-SA 4.0	Adaptor/
FemalePinHolder.sldprt	Solidworks file	CC BY-SA 4.0	Adaptor/
FemalePinHolder.stl	3D-printable file	CC BY-SA 4.0	Adaptor/
Adaptor_PCB.pcb	KiCAD PCB file	CC BY-SA 4.0	Adaptor/

Table 2
Bill of materials summary.

Component	Qty	Total Cost (NZD)
DAQ unit	1	82.63
RJ45 cables	4	39.68
Adaptor unit	3	45.00
Hanging scale	1	12.77
USB cable	1	13.49
Mechanical connectors	20	5.62
Total		199.10

and female connectors, were designed in Solidworks. All original FreeCAD (.FCStd) and Solidworks (.sldprt) drawings, as well as their exported stl files (.STL, ready for import into 3D printing software) are available in the online repository with file locations described in Table 1. Note to access any of the design files in the repository, first navigate to the desired component (eg Software, DAQ, Adaptor etc.), and then navigate to the files tab.

3.2. Electronics:

All PCB design was performed using KiCAD. In total, there are 2 PCB boards, 1 main board and 1 smaller board for the adaptor. The main board is used for the DAQ housing the Arduino. The adaptor board connects to the both female and male connectors, and houses an RJ45 port to send data downstream to the DAQ unit.

3.3. Software and firmware

There are 2 software components for this work. The Arduino_read.ino file which programs the Arduino, and a python application used for the main GUI. The Arduino is used to read data from 4 ADCs, and then send that data over a serial connection via USB. The GUI is used for receiving the serial data from the DAQ. It displays the data in real time and records it into CSV file format.

4. Bill of materials summary

The full bill of materials can be found in the online repository at https://doi.org/10.17605/OSF.IO/TP24R. Table 2 shows a summary of the costs of the main components of the system.

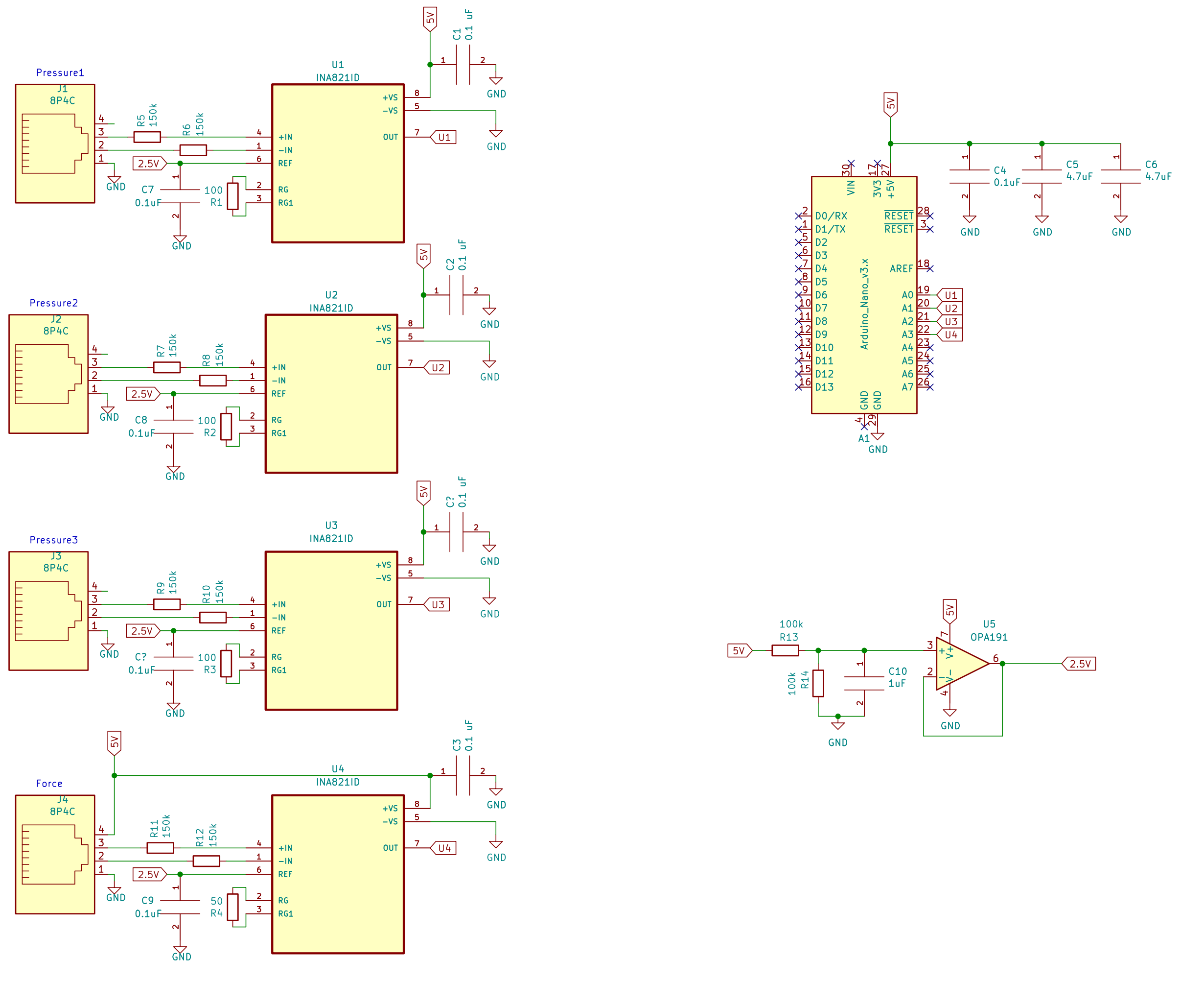

Fig. 1. Schematic diagram for the DAQ.

5. Build instructions

5.1. Assembling DAQ unit

- Place order for PCB. It is possible to use the BOM to order pre-populated PCBs, which is advised.
- If not pre-populated, order all circuit components for the DAQ PCB, as per the BOM.
- Print the DAQ housing base and lid from provided stl files.
- If the PCB was not pre-populated, populate it with components. Note the Arduino should not be soldered directly to the board, but placed on pin headers.

5.2. Assembling adaptors

- Place order for adaptor PCBs. Only 1 adaptor per pressure line is needed up to a maximum of 3 lines (If you are measuring 2 lines, only assemble 2 adaptors)
- Place an order for all required circuit parts (including adaptor PCB, and male and female connector), as per BOM.
- Print the adaptor housing and base, the male and female connectors, and the female connector pin holder (It is advisable to resin print these pieces due to their complexity. If no resin printer is accessible, resin printed parts can be ordered from jlcpcb.com)
- Assemble the male connectors as shown in Fig. 2.

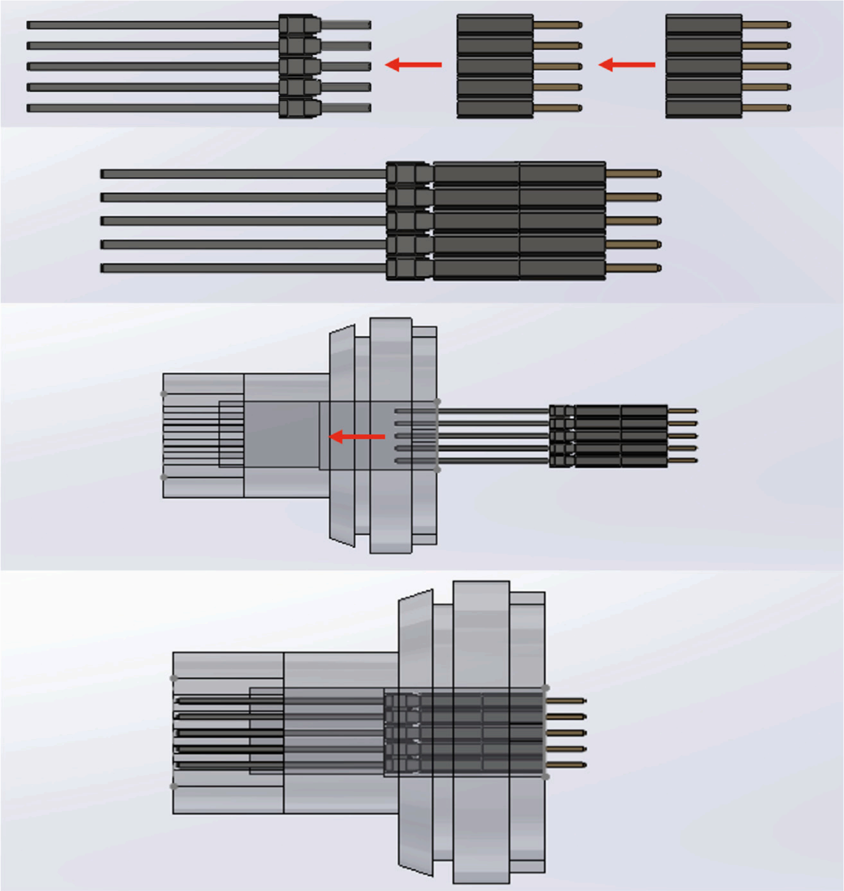

Fig. 2. Diagram for construction of male connector. 1 male to male extended pin header is attached to 2 female to male pin headers (See BOM under Male Connector for exact part numbers). This assembly is then inserted into the small hole on the backside of the male connector such that the small elongated pins of the assembly slide easily through the small gaps on the male connector.

- Insert pins into the female pin holder as shown in Fig. 3. Once completed, insert the pin holder into the female connector as shown in Fig. 4.
- Once completed, the male and female connectors should resemble those of an Edwards truwave connector, shown in Fig. 5.
- Insert the adaptor main PCB into the adaptor housing base and then insert the male and female connectors on each side of the adaptor as shown in Fig. 6, ensuring that electrical connections between the female and male connector, and the main board are properly established. The male connector must be inserted on the right side of the adaptor (when facing the RJ45 port), and the female connector on the left.
- Secure the female and male connector to the adaptor using a strong adhesive. Super glue is not advised. The constructed and fully connected adaptor is shown in Fig. 7

5.3. Assembling hanging scale

- Print housing base and lid for hanging scale

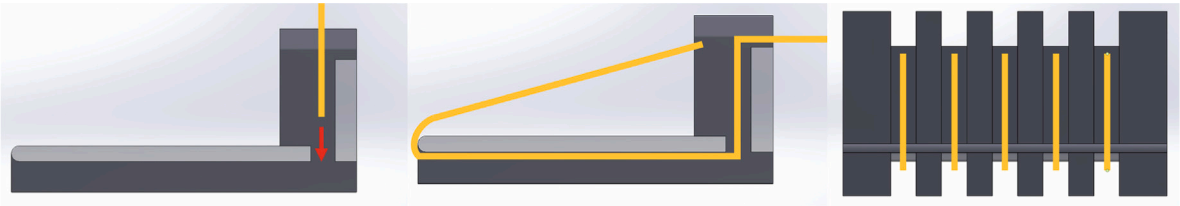

Fig. 3. Diagram for inserting metal pins into female connectors pin holder. The left image shows how to insert the pin, the central image shows how to correctly shape the pin and the right image shows the front view once all pins have been inserted.

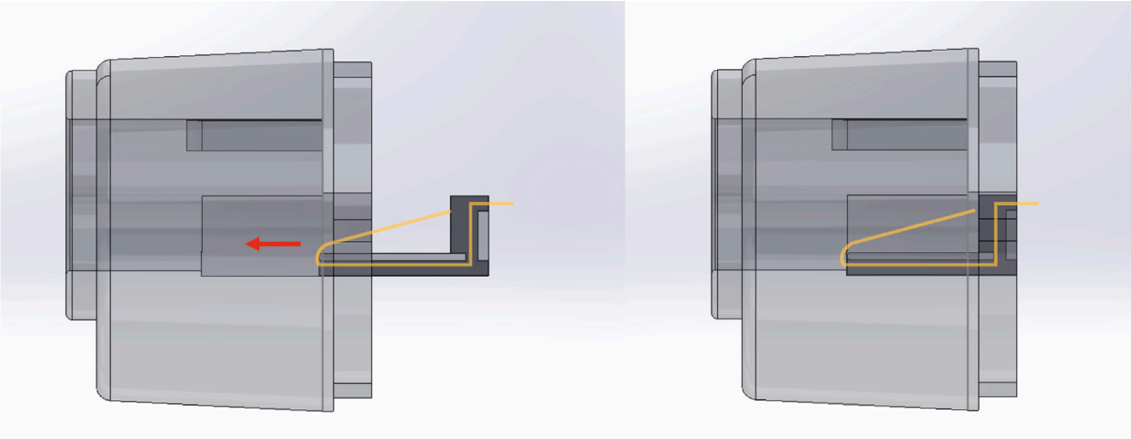

Fig. 4. Diagram for construction of female connector. Pin holder is inserted into the hole on the rear side of the female adaptor as shown. Note the orientation of the female connector.

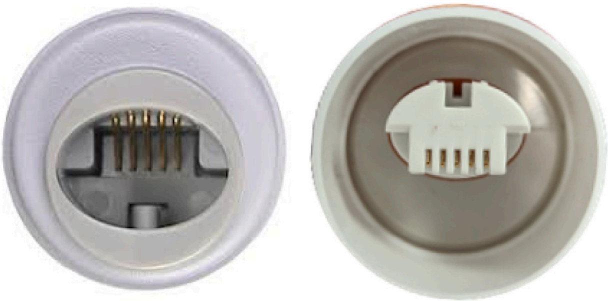

Fig. 5. Edwards truwave male and female connector.

- Assemble hanging scale as shown in Fig. 8.

6. Operating instructions

6.1. Installing software and firmware

- Download source code form online repository. Source code can be found by navigating to the Software component, and opening the files tab. From there, download the Arduino_read.ino file, and either the GUI.zip for the full python code repository, or build.zip for a pre-built windows executable file with all required dependencies.

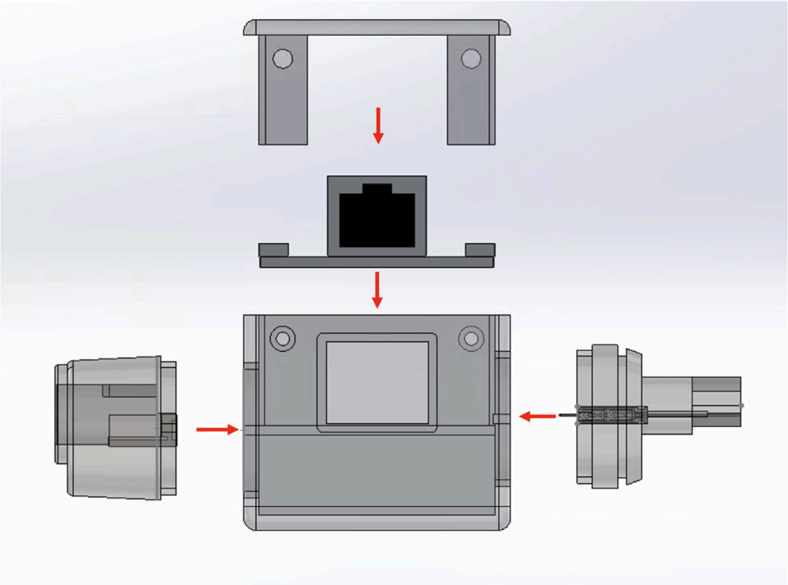

Fig. 6. Pressure adaptor construction diagram. The PCB is inserted into the casing, and then the male and female connectors are attached to the PCB. Finally, the casing is closed with the enclosure lid.

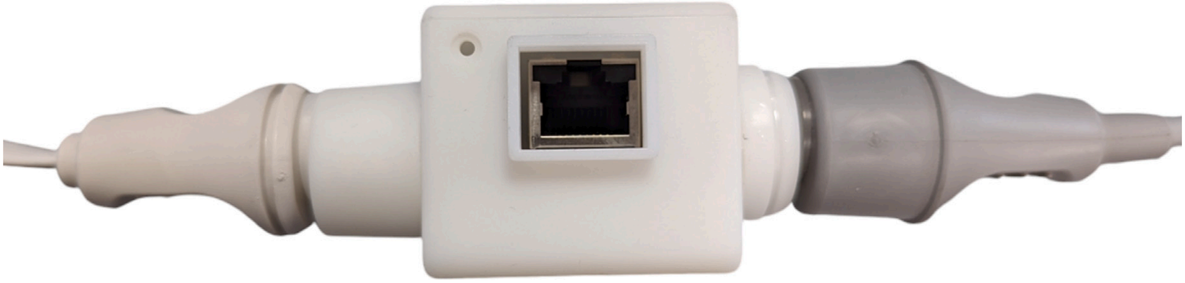

Fig. 7. Fully constructed pressure adaptor.

- Load code, Arduino.ino, onto Arduino board using the Arduino IDE.
- If the full python code repository was downloaded (GUI.zip), a requirements.txt file can be found in the GUI.zip folder. The python dependencies can be installed by opening a terminal, navigating to the extracted GUI folder, and running ‘`pip install -r requirements.txt`’. Once all dependencies have been installed, the software can be run from the terminal by running the command ‘`python main.py`’.

6.2. Connecting devices

- Connect the DAQ to the computer via USB-C cable
- Connect the hanging scale to port 4 of the DAQ via RJ-45 cable. It is important to connect to port 4, as this is the only port which supplies power for the load cell. A diagram of the numbered DAQ ports is shown in Fig. 9.
- Connect the pressure sensor and the bedside monitor to the adaptor
- Connect the adaptor to any of ports 1–3 of the DAQ via an RJ-45 cable. (Port 4 should be avoided as this will result in a connection between VCC of the DAQ and VCC of the bedside machine.)

6.3. Running the system

- With the DAQ connected to the computer, run the GUI, either using the GUI.exe file provided in the build.zip folder, or by running the GUI.py file in the python repository using the command ‘`python main.py`’ from inside a terminal window.

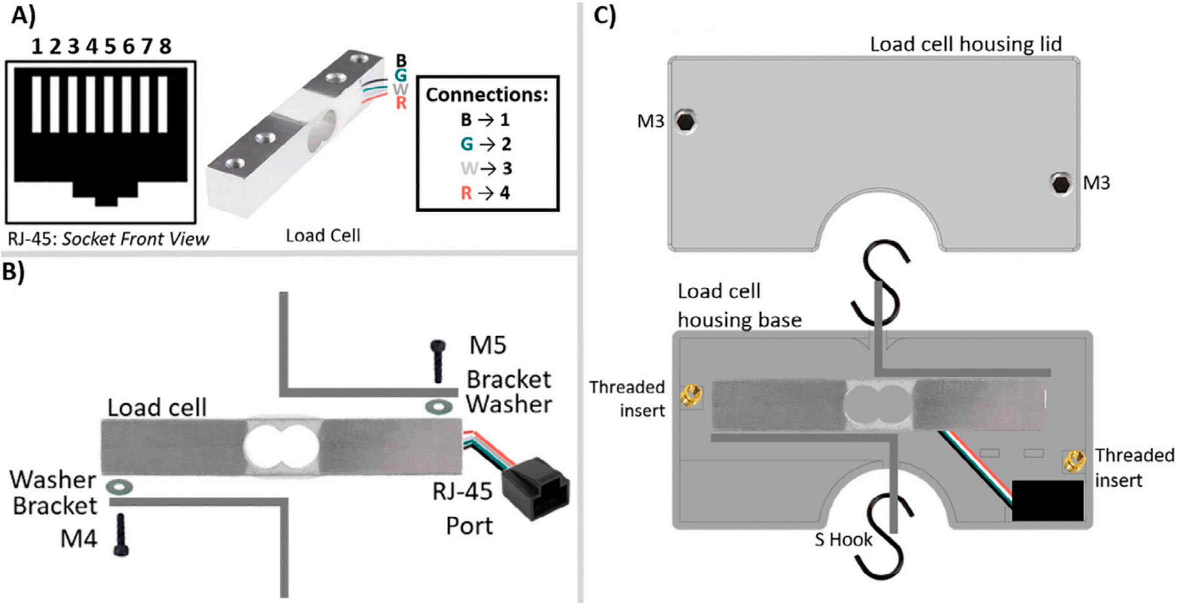

Fig. 8. Hanging scale assembly. Image from [11]. The wires of the load cell are soldered to the RJ-45 port as per A. The load cell is then assembled as shown in B, and inserted into the casing as shown in C.

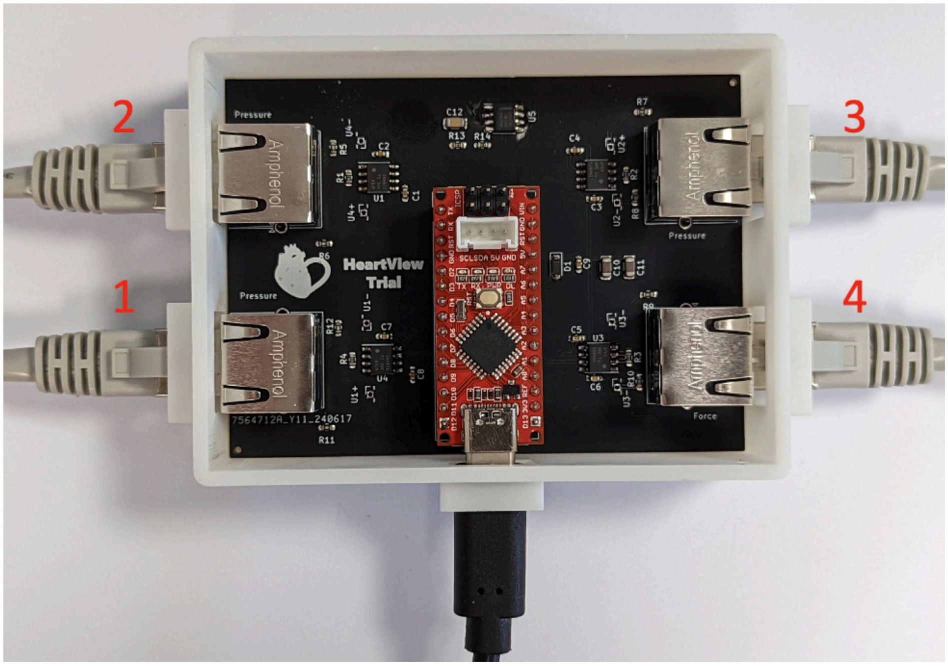

Fig. 9. DAQ unit with port numbers labeled. Ports 1–3 are for pressure sensor inputs, and port 4 is for the load cell input.

The GUI shows 4 plots, 3 for pressure lines, and 1 for the fluid data, labeled as per the axes. Lines from unconnected ports will show large amounts of noise, and can be turned off using the toggle switch next to the graph.

- The data can be calibrated by using the “set-zero” button next to the graphs. To calibrate the pressure data, open the pressure sensor to atmospheric pressure and click the “set-zero” button. The same can be done for the hanging scale. It is advisable

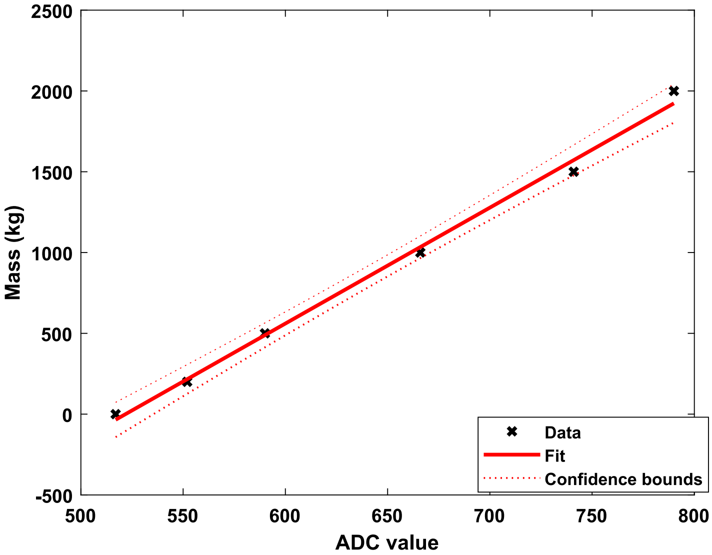

Fig. 10. Hanging scale calibration curve.

Table 3
RMS errors for pressure recordings.

Recording	RMS	Correlation Coefficient
1	1.17	0.9997
2	4.37	0.9934
3	2.90	0.9978
4	2.53	0.9978

to set the hanging scale to 0 while it is already hanging, to account for the weight of the apparatus. Refer to Section 7.1 for further instructions on device calibration if necessary.

- To save data, click on the "record" button, which begins the recording process. To end recording, click on the "stop and save" button. The recorded data will be saved in csv format 1 directory level up in a folder called "Save Trial Data"

7. Validation and characterization

The hanging scale was calibrated using known weights hung from the sensor and obtaining the calibration curve. The calibration curve is shown in Fig. 10, and the linear calibration coefficients were found to be 7.1717 and −3742.4 for the gradient and offset respectively. Note the 5 V amplifier used for amplification of the load cell signal amplifies both positive and negative voltages. Thus an input signal of 0 V, resultant form a 0 kg load, is output as 2.5 V from the amplifier, which is roughly equivalent to an ADC value of 512. Thus, a 0 kg load is read as 512 from the ADC.

The calibration for the pressure sensor was obtained by analyzing data from both the Philips monitor and DAQ unit. A Philips monitor was made available for testing of the device. The Philips monitor used for testing had the capability to export data via a LAN cable to a computer (note this is not usually available on in-hospital units, and even when it is, it does not allow for real time data capture). For the test, the pressure sensor was connected to a syringe which acted as a pressure source. The syringe was used to create several pulses and the data from both the DAQ and the Philips monitor were collected. The Philips monitor records data at 125 Hz, while the DAQ records at 500 Hz.

The Matlab "interp" function was used to interpolate the Philips data to a 500 Hz sample rate. Matlab's cross correlation function "xcorr" was then used to find the sample offset between the two waveforms. Once the waveforms were aligned, the calibration coefficients for the pressure sensor were found by fitting the data from the DAQ to the data from the Philips monitor using simple linear regression. The calibrated DAQ data is shown with the monitor data in Fig. 11. The calibration yielded the linear calibration coefficients of 0.3832 and −199.498 for the gradient and offset respectively.

Once calibration values were found, 4 pressure recordings of 5 s were made from 4 different DAQ units. The syringe driver setup was repeated, and the results between the Philips monitor and DAQ units were compared. The recording are shown in Fig. 12.

A slight difference between recordings can be seen on the largest pressure peak in the fourth block of Fig. 12. This is because the DAQ unit saturates at 200mmHg. While the recordings of the Philips device did not saturate at this level, the Philips monitors only display pressure up to 150mmHg. The RMS errors as well as the correlation coefficients for each of the recordings are shown in Table 3.

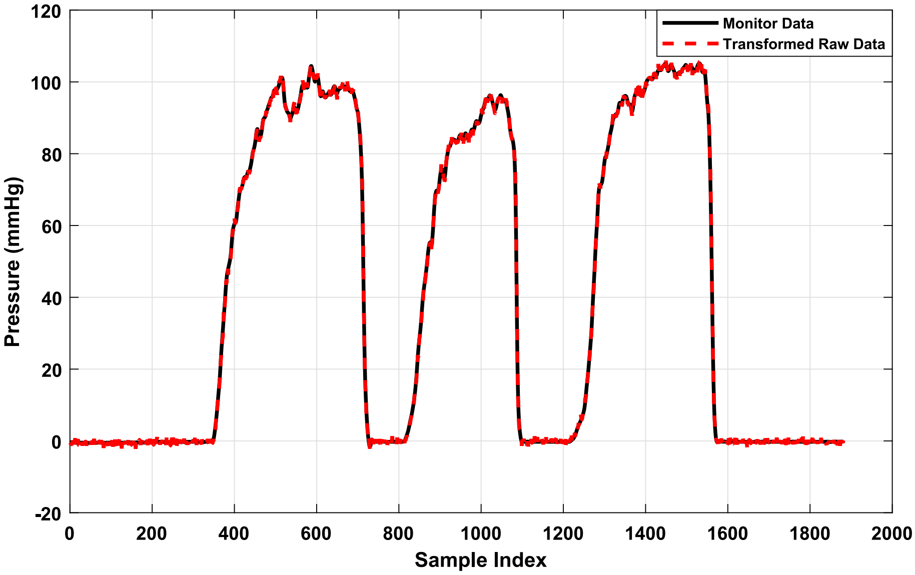

Fig. 11. Monitor pressure data vs calibrated data from DAQ.

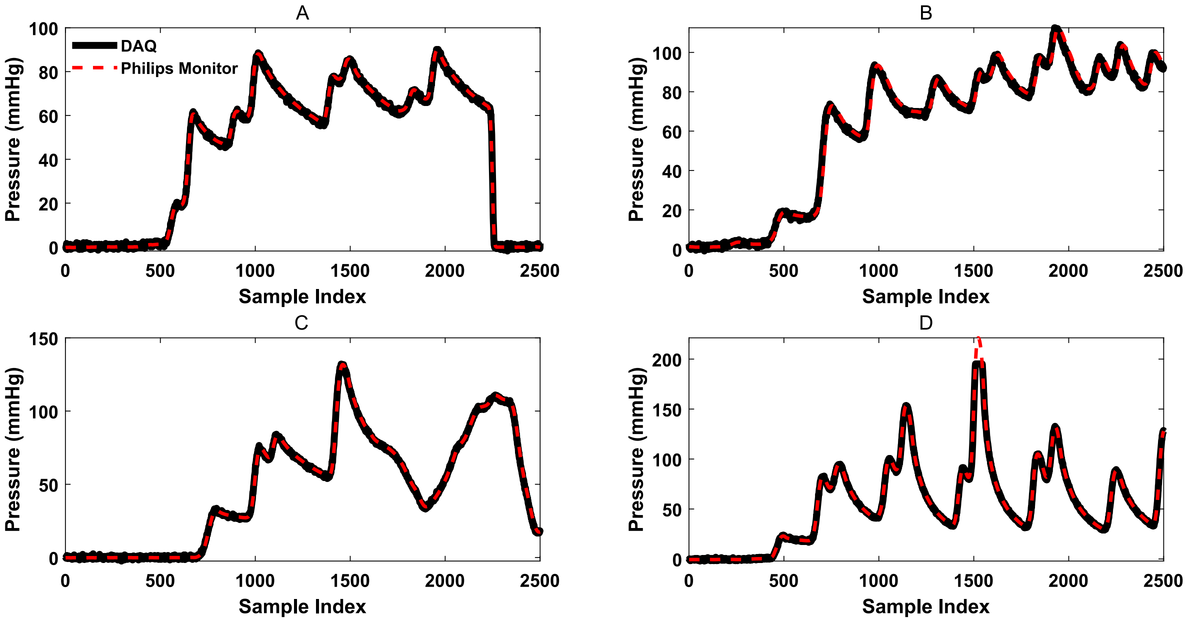

Fig. 12. Pressure recordings from 4 separate DAQ units compared with simultaneous recording from Philips monitor. Subplots A, B, C and D were recorded by DAQ units 1, 2, 3 and 4 respectively.

The RMS range from 1.17 to 4.37, this could be perceived as high, however, the DAQ data is noisy, and the Philips data is filtered, so a slight RMS is expected. Furthermore, the very strong correlation coefficients confirm the devices ability to accurately capture pressure waveforms.

The hanging scales were verified by hanging a variety of known weights on 4 different scales, and ensuring that the GUI displayed the correct reading for each weight.

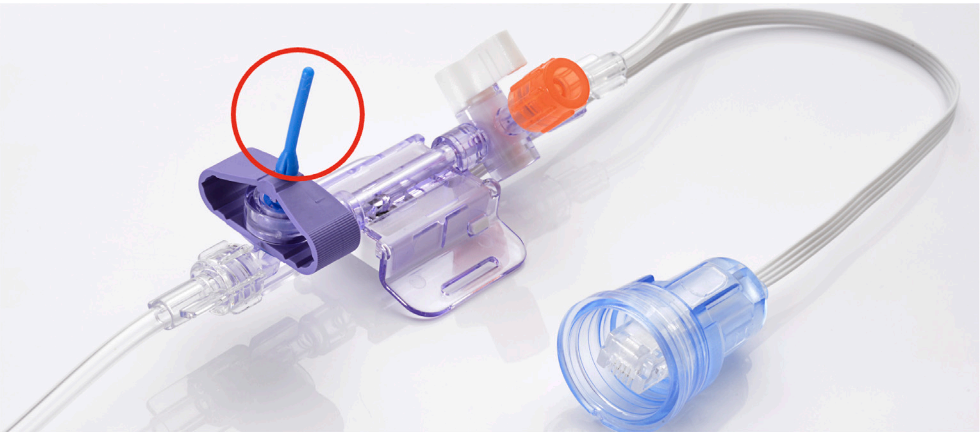

Fig. 13. Pressure sensor with chord used for exposing sensor to atmospheric pressure circled in red.

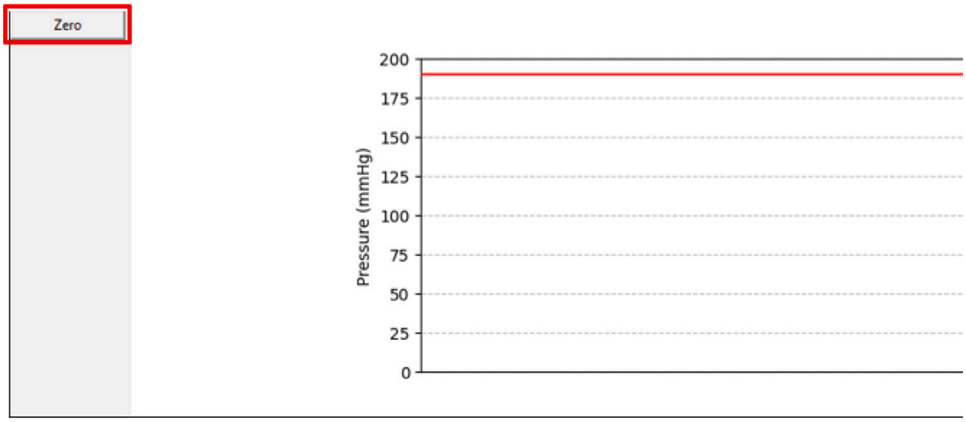

Fig. 14. Zero button for resetting calibration offset. The zero button is boxed in red.

7.1. In-use device calibration

The pressure sensor calibration is expected to change based on the surrounding atmospheric pressure, which is subject to change depending on environmental conditions. This adds a potential offset error to the data. In order to combat this, the same technique which is used for the Philips bedside machines is used for calibration of the DAQ data. On the python GUI, there is a built in “zero” button, which resets the calibration offset to the current atmospheric conditions. To use it, the pressure sensor must first be exposed to atmospheric pressure, and then the zero button pushed. The pressure sensors have a built in chord, which when pulled, opens a valve and exposes the sensor to atmospheric pressure. This is the same process that clinicians and bedside nurses are accustomed to calibrating pressure sensors attached to the Philips bedside monitor. The process thus adds little to no complexity to the existing bedside calibration procedure. Figs. 13 and 14 show the pressure sensor chord being pulled and the zero button on the python GUI respectively. Pressure sensors are expected to drift over long lifetimes. However, hospital pressure sensors are disposable and drift is not expected for single use.

Likewise, the loadcell offset is expected to change based on the weight of the apparatus used for hanging the fluid bag from it. This is addressed in a similar manner, with a “zero” button added which can be pressed once the hanging apparatus has been installed.

7.2. Device potential and future use

A novel, cost-effective data capture system has been created for recording of haemodynamic measurements in the ICU, as well as fluid infusion data. The device uses a simple Arduino nano microcontroller which connects to a computer via USB and displays and records data via a GUI interface. Multiple devices have been created, tested and validated, and are awaiting future use for small scale hospital trials. This device could be useful to researchers who require but do not currently have access to both real time or retrospective haemodynamic measurements from the ICU. This data could be highly useful for any of the following purposes:

- Low cost patient monitoring
- Cardiovascular modelling
- Haemodynamic trend analysis

Future improvements for the device would be to remove the need for a computer, using an onboard screen and SD card, as well as adding opto-isolation for all pressure lines. This would make the overall system far more portable, and easier to use regularly in an ICU setting.

CRediT authorship contribution statement

James Cushway: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Formal analysis, Conceptualization. **J. Geoffrey Chase:** Supervision, Resources, Conceptualization. **Thomas Desaive:** Supervision, Resources. **Liam Murphy:** Conceptualization. **Isaac L. Flett:** Methodology. **Geoffrey M. Shaw:** Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Funding: This work was supported by EU H2020 ERA Permed JTC2021, “Personalised perfusion guided fluid therapy”.

Funding: The project was supported by EU H2020 R&I programme (MSCA-RISE-2019 call) under grant agreement #872488 — DCPM.

References

[1] R. Kumar, P. Dubey, A. Zafer, A. Kumar, S. Yadav, Past, present and future of blood pressure measuring instruments and their calibration, Measurement 172 (2021) 108845, http://dx.doi.org/10.1016/j.measurement.2020.108845, [Online]. Available.

[2] C. Lockwood, T. Conroy-Hiller, T. Page, Vital signs., JBI Libr. Syst. Rev. 2 (2021) 1–38, http://dx.doi.org/10.11124/jbisrir-2004-371, [Online]. Available.

[3] J. Cushway, L. Murphy, J.G. Chase, G.M. Shaw, T. Desaive, Physiological trend analysis of a novel cardio-pulmonary model during a preload reduction manoeuvre, Comput. Methods Programs Biomed. 220 (2022) 106819, http://dx.doi.org/10.1016/j.cmpb.2022.106819, [Online]. Available.

[4] L. Murphy, S. Davidson, J.L. Knopp, J.G. Chase, T. Zhou, T. Desaive, State analysis of total stressed blood volume and arterial elastance during induced sepsis, in: 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC, 2019, pp. 2951–2954, http://dx.doi.org/10.1109/EMBC.2019.8857518, [Online]. Available.

[5] A. Pironet, T. Desaive, J. Geoffrey Chase, P. Morimont, P.C. Dauby, Model-based computation of total stressed blood volume from a preload reduction manoeuvre, Math. Biosci. 265 (2015) 28–39, http://dx.doi.org/10.1016/j.mbs.2015.03.015, [Online]. Available.

[6] R. Enev, P. Krastev, F.A. and, Prediction of fluid responsiveness: a review, Biotechnol. Biotechnol. Equip. 35 (1) (2021) 1147–1155, http://dx.doi.org/10.1080/13102818.2021.1960190, [Online]. Available.

[7] F. Guarracino, P. Bertini, M.R. Pinsky, Cardiovascular determinants of resuscitation from sepsis and septic shock, Crit. Care 23 (1) (2019/04/15) 118, http://dx.doi.org/10.1186/s13054-019-2414-9, [Online]. Available.

[8] A. Pironet, Model-Based Prediction of the Response to Vascular Filling Therapy, University of Liège, 2016.

[9] J. Cushway, L. Murphy, J.G. Chase, G.M. Shaw, T. Desaive, Modelling patient specific cardiopulmonary interactions, Comput. Biol. Med. 151 (2022) 106235, http://dx.doi.org/10.1016/j.compbiomed.2022.106235, [Online]. Available.

[10] N.A. Davey, L. Murphy, J.G. Chase, C. Zhou, Model parameter identification as an index of fluid responsiveness, IFAC-PapersOnLine 58 (24) (2024) 584–589, http://dx.doi.org/10.1016/j.ifacol.2024.11.102, [Online]. Available. 12th IFAC Symposium on Biological and Medical Systems BMS 2024.

[11] R. Smith, A. Rolfe, C. Cameron, G.M. Shaw, J.G. Chase, C.G. Pretty, Low cost circulatory pressure acquisition and fluid infusion rate measurement system for clinical research, HardwareX 11 (2022) e00318, http://dx.doi.org/10.1016/j.ohx.2022.e00318, [Online]. Available.

James Cushway received his B.Sc. eng (hons) specializing in Electrical and Computer Engineering from the University of Cape Town in 2017. He is currently a Ph.D. candidate at the University of Canterbury (2021-present). His research focuses on using engineering and computational methods to improve healthcare and our understanding of physiology, specifically, the cardiovascular and respiratory systems. The developed model-based systems aim to improve crucial decision making in the ICU, ultimately improving patient outcomes.