HHS Public Access

Author manuscript

HardwareX. Author manuscript; available in PMC 2019 June 12.

Published in final edited form as:

HardwareX. 2019 April ; 5: . doi:10.1016/j.ohx.2019.e00063.

All-in-one automated microfluidics control system

Craig Watson and **Samuel Senyo***

Department of Biomedical Engineering, Case Western Reserve University, Cleveland, OH, United States

Abstract

We present a fully-integrated solution for controlling pneumatically-driven microfluidic chips, featuring a pump, one or more pressure regulators and up to 32 solenoid valves, controlled by a microcontroller. The microfluidics control system requires only a power source and a computer or mobile device for its operation. A touchscreen interface communicates with the microcontroller over USB or Bluetooth and allows users to control the system with ease either manually or autonomously, allowing experiments to run with no user intervention. The pressure regulators were purpose-built, enabling integrated pressure sources on-board, rather than relying on external equipment. These regulators can also be used as stand-alone devices in any other application.

Keywords

Solenoid valve; Pressure regulator; Pressure controller; PDMS; Arduino; ESP32

1. Hardware in context

Pneumatically-driven microfluidic chips can require the connection of dozens of tubes to supply reagents and to actuate on-chip elastomeric valves. This requires a set-up capable of delivering pressurized air at a range of different pressures, split into many lines which must be individually controlled. Common approaches are to either using a purpose-built, manually operated pneumatic apparatus, which requires considerable technical knowledge to assemble, or with commercial control systems, which can cost tens of thousands of dollars [1–3]. To address this, we have developed an open-source, all-in-one microfluidics control system that includes pressure sources and regulators, and up to 32 solenoid valves to control any pneumatically-driven microfluidic chip at a fraction of the cost of a commercial control system.

The main functionality of pneumatic microfluidics controllers – commercial or DIY – is the actuation of multiple valves to switch pressure independently across multiple lines of tubing

*Corresponding author. ssenyo@case.edu (S. Senyo).

Declaration of interest

None.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ohx.2019.e00063.

(which are connected to a microfluidic chip). The use of computer-controlled solenoid valves provides a way to automate most aspects of an experiment, and several such devices have been reported in recent years [4–7].

Our design builds upon existing solutions and adds on-board pressure generation and regulation, features that are rarely included in comparable microfluidic controllers. Pressure is commonly provided through centralized compressed air, or compressed air tanks, and regulated using manual pressure regulators. Such systems rely on bulky equipment and necessitate considerable infrastructure.

Integrating a pump and pressure regulators into the system allows the construction of a portable, all-in-one microfluidics control system, which needs only a power source to function. Not only does this simplify the setup for the user, it also makes it easier for labs with limited resources to explore the use of microfluidics for their research. Fig. 1 shows the control system as used in our lab, connected to a microfluidic chip and operated via a laptop.

Compared to similar open source microfluidics controllers, our system is compact and occupies little space on the bench top, as all valves are integrated into the housing. The increased integration comes at the expense of some flexibility in valve configurations, as our system requires assembling connectors for the valves, rather than making use of screw terminals. Other systems may also allow the connection of more than 32 valves [4,6].

The pressure regulators detailed here have applications beyond the field of microfluidics, as they are compact, precise, consume little air, and offer multiple digital and analog interfaces for control. We anticipate they would be useful to fields such as soft robotics or any other application requiring on-board pressure regulation.

2. Hardware description

2.1. Main PCB

The system was designed with modularity in mind, and as such can support various hardware configurations. The components in the bill of materials reflect our current configuration, but there is some flexibility in the choice of most items.

The main printed circuit board (PCB) was designed as an expansion board for an Espressif ESP32 devkit C – a popular development board for wireless embedded devices which features WiFi, Bluetooth (Classic and Low Energy), analog inputs and outputs, i^2c, and more. It is also Arduino compatible.

The main PCB consists mainly of an I/O expander and power drivers, featuring 5 ULN2803 Darlington arrays: 4 of these can each drive 8 inductive loads at up to 500 mA, and the last is wired to drive two loads at up to 2A each. Functionally, the first four Darlington arrays are used to actuate up to 32 solenoid valves, and the last controls up to two pumps. While our implementation features Pneumadyne 10 mm solenoid valves, any 12 V solenoid valves can be used, as long as each of them draws less than 500 mA, and a total of less than 3.2A for all valves (higher currents may be allowable, but the PCB was designed and tested with an overall consumption of 100 mA per valve in mind).

This PCB also provides an interface for pressure regulators. These were first designed for commercially-available regulators (Parker OEM series), which are powered by 24 V and use analog pins to specify the setpoint and current pressure. Voltage dividers and amplifiers are used to convert between 0 and 3.3 V and 0–5 V signals, while an external step-up converter provides the 24 V supply. We later developed our own pressure regulators, which offer more precise pressure control and measurement, as well as multiple available interfaces (analog, i^2c and USB), all at less than half the price of the commercial parts they replaced. The main PCB supports up to two analog pressure regulators, and multiple digital regulators. Only one connector is provided for these digital regulators, as they can be daisy-chained together via their i^2c interface.

2.2. Pressure regulators

The pressure regulators follow a simple design (Fig. 2). A proportional solenoid valve is placed on the inlet, and the outlet of this valve forms the outlet of the regulator, as well as being connected to a pressure sensor and a second proportional solenoid valve that can vent to atmospheric pressure. A microcontroller (ATmega32u4) implements a closed-loop control algorithm to hold the output pressure at the requested level, opening the inlet valve to increase pressure or opening the vent valve to lower pressure (or the inverse, if the supply is a vacuum pump). Our dual-valve design improves upon the commercial alternative [8], which uses an always-open vent to relieve excess pressure, and allows a limited pressure source such as a small diaphragm pump to be used, as little to no air is unnecessarily vented.

The pressure regulators were designed as stand-alone products rather than being integrated into the main PCB, mainly to maximize modularity of the system and to make it as easy as possible for others to build and use them in their own projects. Additionally, the size increase from adding a separate PCB for each regulator is minimal, and allows any number of regulators to be used at once. Each regulator has its own microcontroller and needs only a power supply and a given setpoint to operate. The setpoint and current measured pressure can be communicated over analog pins, i^2c or USB.

Using the i^2c interface, several regulators can be daisy-chained. This interface also offers a useful extra feature, where the regulators will signal if and when the supply pressure is estimated to be too low to maintain the current setpoint. In our application, this allows the main microcontroller to switch the pump on only when it is needed, which helps to prolong pump lifetime and reduce noise and power usage. In addition, with the USB and i^2c interfaces, the parameters of the PID control loop can be updated in real time without the need to reprogram the microcontroller. This is especially useful for development purposes.

3. User interface

The user interface was developed using Qt cross-platform libraries. It has been tested on Windows, Linux and Android, and could be ported to macOS and iOS. The interface and microcontroller communicate via a simple serial protocol, supporting both USB and Bluetooth. The interface has two main panels. The manual control panel displays the status of all valves and pressure regulators, and allows the user to set the state of each of these

components. The routine panel allows the user to load a routine, or experiment, to run commands autonomously.

Routines are human-readable text files, with one instruction per line. Instructions take the following form:

pressure 1 20 psi

psi valve 16 open

wait 2 minutes

valve 16 close

Using this format, it is possible to run an experiment with no user intervention, and no programming knowledge is necessary to define or run the routine. In the software, the user simply selects the routine file to be run; the file is checked for errors, and can then be executed. The current status is then shown, with the current step highlighted.

The user interface is based on C++ and QML code. It was designed to be easy to customize and modify, with minimal programming experience necessary to change the setup of pressure regulators, valves and other components. Documentation specific to this software is included in the *ufcs_pc_software.zip* file.

4. Design files

4.1. Design files summary

Design file name	File type	Open source license	Location of the file
Design file 1	e.g., *CAD files, figures, videos*	*All designs must be submitted under an open hardware license. Enter the corresponding open source license for the file.*	*Enter a link to the online location or the sentence: "available with the article", as appropriate*
bill_of_materials.xlsx	Excel spreadsheet	Creative Commons Attribution 4.0	https://osf.io/byvmh
ufcs_pcb_design_files.zip	PCB files, KiCAD	Creative Commons Attribution 4.0	https://osf.io/d3rqa
ufcs_pcb_gerber_files.zip	PCB files, gerber	Creative Commons Attribution 4.0	https://osf.io/z2hec
ufcs_baseplate.zip	CAD files	Creative Commons Attribution 4.0	https://osf.io/vrthp
ufcs_microcontroller_software.zip	C++ code	GPL v3	https://osf.io/e68xp
ufcs_pc_software.zip	C++/QML code	MIT License	https://osf.io/vb9nr
pressurereg_software.zip	C++ code	MIT License	https://osf.io/tr2d6
pressurereg_pcb_design_files.zip	PCB files, KiCAD	Creative Commons Attribution 4.0	https://osf.io/mfc2b
pressurereg_pcb_gerber_files.zip	PCB files, gerber	Creative Commons Attribution 4.0	https://osf.io/xdsbn
pressurereg_manifold.zip	CAD files	Creative Commons Attribution 4.0	https://osf.io/7n5yt

Design file name	File type	Open source license	Location of the file
pressurereg_enclosure.zip	CAD files	Creative Commons Attribution 4.0	https://osf.io/apmuk
ufcs_laser_cut_case.zip	DXF files	Creative Commons Attribution 4.0	https://osf.io/hmw7s
pump_mount.zip	CAD files	Creative Commons Attribution 4.0	https://osf.io/pqdfn

- bill_of_materials.xlsx: Bill of materials for the system. The first sheet lists components of the overall system; the second sheet details the components of the main PCB; the third sheet details the components of the pressure regulators; and the last sheet lists tubing and fittings necessary to connect all pneumatic components.
- ufcs_pcb_design_files.zip: KiCAD design files for the main PCB.
- ufcs_pcb_gerber_files.zip: Gerber files for the main PCB.
- ufcs_baseplate.zip: CAD files for the baseplate. Includes Autodesk Inventor part file and mechanical drawing.
- ufcs_microcontroller_software.zip: Software source code for the ESP32.
- ufcs_pc_software.zip: Source code for the user interface software.
- pressurereg_software.zip: Source code for the pressure regulator software.
- pressurereg_pcb_design_files.zip: KiCAD design files for the pressure regulator PCB.
- pressurereg_pcb_gerber_files.zip: Gerber files for the pressure regulator PCB.
- pressurereg_manifold.zip: CAD files for pressure regulator manifold. Includes Autodesk Inventor part file, STL file and mechanical drawing for machining.
- pressurereg_enclosure.zip: CAD files for pressure regulator enclosure. Includes Autodesk Inventor part file and STL file for 3D printing.
- ufcs_laser_cut_case.zip: DXF files that can be used to laser cut a case for the control system out of acrylic.
- pump_mount.zip: CAD files for a pump mount. Includes Autodesk Inventor part file and STL file for 3D printing.

5. Build instructions

5.1. Main PCB assembly

- Solder components onto the PCB as in Fig. 3. Start with SMD components, then add connectors and other through-hole components.
- We recommend soldering 2-row female header connectors to the PCB, in which the ESP32 board can be placed. Every pin is doubled, making it easy to plug in

jumper wires for debugging purposes or for adding functionality. Alternatively, the ESP32 board can be soldered directly to the PCB.

5.2. Pressure regulator PCB assembly

- Solder components onto the PCB as in Fig. 3. Start with SMD components, then add through-hole components and connectors.
- The PCB supports either analog or SPI pressure sensors. Different pins are used for either kind of sensor; follow the markings on the PCB based on the specific sensor chosen.
- Additionally, either 3.3 V or 5 V pressure sensors can be used. Based on the supply voltage of the sensor, solder the appropriate pads of the J5 jumper together. For example, for a 5 V sensor, add a blob of solder between the top and middle pad.
- If a 5 V analog sensor is used, populate R5 and R6 according to the bill of materials. If using a 3.3 V analog sensor, do not add resistors but instead add a blob of solder to short the two pads of R5 together. If using an SPI sensor, skip this step as these connections are unneeded.

5.3. Pressure regulator programming

The pressure regulator uses an Atmega32u4 microcontroller, which needs to be flashed with an Arduino-compatible bootloader before any code can be uploaded to it. This can be done with a specialized programmer, or by using an Arduino as a programmer. The ESP32 can also be used as a programmer, avoiding the need for extra equipment. To do so, follow the instructions below.

- Extract the pressure_regulator_software.zip archive.
- Download and install the Arduino IDE, and open it.
- Install the ESP32 and Sparkfun Pro Micro board definitions:
- Open File > Preferences. Under “Additional Board Managers URL”, add https://dl.espressif.com/dl/package_esp32_index.json and https://raw.githubusercontent.com/sparkfun/Arduino_Boards/master/IDE_Board_Manager/package_sparkfun_index.json (one address per line).
- Open the Board Manager in Tools > Boards > Board Manager. . .
- Select “ESP32” and click Install.
- Select “Sparkfun AVR board” and click Install.
- Select Tools > Board > ESP32 Dev Module.
- Select File > Open. . . and open *ArduinoISP.ino*, located in *pressure_regulator_software/res/*.
- Plug in the ESP32 via USB; select the right port under Tools > Port.
- Click the Upload button to upload the ArduinoISP code to the ESP32.

- The ESP32 now acts as a programmer, to flash the Arduino bootloader to the pressure regulator board. To do so, connect the IO19, 3.3 V, IO18, IO23, IO16, and GND pins on the ESP32 to the ICSP header on the pressure regulator board, as in Fig. 4.
- Select Tools > Board > Sparkfun Pro Micro.
- Under Tools > Processor: select Atmega32U4 (3.3 V, 8 MHz).
- Under Tools > Programmer: select “Arduino as ISP”
- Select Tools > Burn Bootloader.

The pressure regulator will now be recognized as an Arduino-compatible board, and the software can be uploaded to it. To do so, we recommend using the PlatformIO IDE.

- Download the PlatformIO IDE from platformio.org/install and install it.
- Open the project, located in *pressure_regulator_software*.
- If necessary, adjust the pressure bounds based on the sensor you selected by editing the values of the *minPressure* and *maxPressure* constants at lines 36–37 of *src/main.cpp*.
- Connect the pressure regulator to your computer via USB, and click Upload in PlatformIO to build and upload the project.
- Optionally, to test that the pressure regulator has been programmed successfully, open a serial monitor. The regulator should output lines of six comma-separated numbers several times per second, indicating the setpoint, current pressure, K_p, K_i, K_d and PID output.

5.4. ESP32 programming

- Open PlatformIO, and load the project located in ufcs_microcontroller_software.
- Connect the ESP32 to the computer via USB, and click Upload in PlatformIO to build and upload the project.

5.5. Pressure regulator assembly

- Machine the manifold following the drawing (from *pressure_reg_manifold.zip*), or have it produced at a machine shop. Alternatively, it can be 3D printed. However, the solenoid valves can heat up considerably, so care must be taken with material choice if 3D printing. The aluminum manifold acts as a heatsink, avoiding thermal issues.
- Secure the solenoid valves to the manifold using M2 screws.
- Using M3 standoff screws, secure the PCB above the manifold. Connect the pressure sensor to the manifold using $1/16^{00}$(1.6 mm) ID (inner diameter) tubing and a $1/16^{00}$ (1.6 mm) barb – 10–32 UNF fitting on the manifold.
- Optionally, 3D print an enclosure (from *pressure_reg_enclosure.zip*) and secure it on top of the PCB using M3 standoffs and flat head M3 screws.

5.6. Control system assembly

- Assemble the solenoid valves onto the manifolds and add fittings for the manifold inlets and each valve outlet.
- Assemble connectors for the solenoid valves. The connectors have 10 pins: 2 for the common 12 V, and one for the negative (black) wire of each valve. Solder the red wires together, grouping 4 together with one extra wire that will be crimped in the connector. Fit these 10 wires into the connector, paying attention to the orientation of the connector. Pins 1 and 2 are used for the 12 V supply, and 3 through 10 for the black wires of each valve. Repeat this process for each group of 8 valves.
  - An alternative to assembling these connectors is to solder female header connectors to the PCB in place of the male receptacles. Any sufficiently rigid wire can then be plugged into the female headers to connect each valve. Note that it will still be necessary to group the common (12 V, red) wires of all the valves together.
- Assemble connectors for the pump. Solder two wires to the contacts on the motor, and add a connector to the other end of the wires. Polarity does not matter for the pump listed in the bill of materials.
- Add fittings to the pressure regulator manifold(s). 1/8″ (3.2 mm) ID tubing is recommended, but smaller or larger sizes can also be used.
- Connect the filter/muffler to the inlet of the pump, and the check valve to the outlet, using 1/8″ (3.2 mm) ID tubing and paying attention to orientation of both parts. Connect the other side of the check valve to the inlet of the pressure regulator. If using more than one pressure regulator, the line can be split to connect the pump to several regulators.
- Secure all the components to a base plate. A CAD file and drawing of a base plate is included; holes can be patterned with a CNC mill, waterjet cutter, laser cutter, or simply a drill or drill press. Aluminum is recommended for the base plate as it will act as a heatsink, but we have also used PMMA with no issue. The holes in the base plate are all either M3×0.5 threaded or M3 clearance holes. M3 screws and standoffs should be used to secure the components in place.
- Optionally, mount the baseplate and all components within a case. The case can be custom-built, or purchased and modified to add cutouts to the faces. Designs for the cutouts that we made are not included, as they will depend on the user's exact choice and placement of components, and chosen method of fabrication. We include an enclosure design that can be laser cut out of 3 mm PMMA (*ufcs_laser_cut_case.zip*), as an alternative to purchasing and modifying one.
- Connect the pump, solenoid valves and pressure regulators to the appropriate connectors on the PCB.

5.7. Software installation

Install the user interface. A pre-built installer is provided for Windows. Alternatively, build the software from source following the readme.md file in the software folder. For Android, an APK package is provided, or can be built in Qt Creator provided that the prerequisite Android build tools are installed.

6. Operation instructions

Power the system on by connecting it to a 12 V, 5A (or more) supply. If using a Windows PC for the user interface, connect the computer to the microcontroller by USB. If using a smartphone, ensure that Bluetooth is switched on. Open the software, and click on “Reconnect”, at the bottom of the screen. Connection typically takes less than 3 s via USB; the first connection via Bluetooth can take up to a minute, but is short on subsequent connections. The status message will update upon connection, and the state of on-screen valve buttons may change as their current (default) state is reported by the microcontroller.

The user interface is split into three screens: Manual Control, Routines, and Log. The Manual Control screen shows the state of each valve and pump, the setpoint and current pressure of the pressure regulators, and the connection status of the microcontroller. Valve and pump states, and pressure regulator setpoints can be set manually by clicking the appropriate buttons or moving the sliders. In the system’s default configuration (as described in this article, with 2 digital pressure regulators), pump 1 is automatically switched on only when needed, based on feedback from the pressure regulators. In this case, manual control of the pump is usually not needed.

The Routines screen handles automated control of the system. Click Browse to select a routine file (an example is provided in *ufcs_pc_software/res/test_routine.txt*. The routine file is scanned for any errors, and these are displayed. Test_routine.txt contains various errors as examples. The user can then either cancel or opt to still run the routine, in which case incorrect instructions are simply skipped. During execution of the routine, the list of steps is displayed, with the current step in bold. If checked, the “run continuously” switch at the top of the list will re-run the routine when it stops.

Finally, the Log screen displays a list of debug, information, warning and error messages. This output is also saved to a file, in the operating system’s default application data path (C:\Users\<username\AppData\Local\ufcs-pc on Windows 10).

7. Validation and characterization

Video 1 shows a demonstration of the microfluidics control system, as it would normally be used in the lab. As shown, pressure can be adjusted for each regulator; all valves can be actuated independently; pressure ranges are suitable for both generating flow and actuating elastomeric valves in a PDMS chip.

Quantitatively, most of the parameters depend on the specific components used. The solenoid valves used in our system have a total cycle time (on/off) of 18 ms. However, these

can be substituted for 12 V solenoid valves from different manufacturers, which will have different cycle times, effective orifices, etc.

Similarly, the pump used here outputs up to 32 psi (220 kPa) at the low flow rates used in our system; if different pressure ranges are required, the pump can be replaced by a different one. For example, a smaller and quieter pump could be used if these relatively high pressures are not required. For pumps and valves, refer to the manufacturers' datasheets for information regarding their performance.

7.1. Pressure regulation

We characterized the performance of the pressure regulators by applying sudden changes in setpoints (step inputs) from 0 to between 10 and 100% of their maximum range, and recording the pressure at the outlet using a separate pressure sensor and microcontroller. Data for the 0–29.5 psi (0–203 kPa) pressure regulator is shown in Fig. 5 and Table 1.

The dynamics of the pressure regulators depend largely on what is connected to their outlet. In the tests reported in Fig. 5 and Table 1, semi-rigid 1/8″ (3.2 mm) inner diameter tubing was attached and sealed, in order to approximately match the dead volume (in the tubing, connectors and valve manifolds) of the setup as it is used in our lab, and to approximate the negligible flow rates on the outlet of the solenoid valves when tubing is connected to a microfluidic chip. Actual performance may vary depending on the inlet pressure applied and on the downstream connections.

For these tests, the PID control loop ran every 1 ms, with parameters $K_p = 0.4$, $K_i = 0.3$ and $K_d = 0$. These parameters were set empirically, and could be tuned further to improve the performance of the control loop. Our main concern when tuning the parameters was to minimize steady-state error; less importance was given to lowering overshoot, as we tend to adjust pressure incrementally and not in large steps, in practice. Further work to improve the response could include modeling the dynamics of the system, to better design a controller (not necessarily PID) for more robust, precise pressure regulation.

7.2. Power consumption

Power consumption of the system is an important consideration, especially if it is used in portable applications. Table 2 details the power consumption, broken down by components. The main power draw comes from the pump and solenoid valves, which require 12 W and 1.3 W each, respectively. In our application, up to 32 valves are used simultaneously, requiring a theoretical maximum of 41.6 W for the valves alone. However, these valves are kept open a majority of the time. Using normally open valves therefore decreases power consumption substantially, as these valves consume no power when in their open state. The main factors for users to consider with regards to power draw are the number and type (normally open or normally closed) of valves.

The pressure regulators were designed to minimize air loss and thus do not need the pump to be on constantly in order to maintain pressure. When pressure is sufficiently high, as communicated by the pressure regulators to the main microcontroller, the pump is switched off. Air is lost mainly through cycling the solenoid valves, and leaks in tubing and

connections. Actual duty cycle of the pump will therefore depend on these factors as well as the pressure requested, and will vary depending on the application.

In conclusion, the microfluidics control system provides a practical, affordable all-in-one solution to controlling microfluidic chips. In comparison with similar existing open-source systems, this setup offers a compact footprint and high potential for automation, thanks to the integration of computer-controlled pressure sources. The custom-designed pressure regulators improve upon both open-source and commercial equivalents, offering high accuracy in an efficient, embedded package. A notable limitation of the system is its relatively complex assembly, which requires experience with surface-mount soldering (for the PCB assembly) and, potentially, with machining (for the pressure regulator manifold). However, it is possible to order PCBs fully-assembled, and to outsource production of the manifolds to a machine shop, making the assembly considerably easier in exchange for a higher cost. The flexibility in valve and pump choice also make it trivial to adapt the system to different end user needs.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgements

This work was supported by the Case Western Reserve University Faculty Investment Fund (RES221997) and the National Institutes of Health (NIH 1 C06 RR12463–01).

References

[1]. Microfluidics Pneumatic Controller – ADEPT | ALine. Available at: https://alineinc.com/microfluidics-products-for-sale/controller-instrumentation/pneumatic-controllers/adept-12-channel/. (Accessed: 10th December 2018).

[2]. Elveflow microfluidic flow control – Elveflow. Available at: https://www.elveflow.com/microfluidic-flow-control-products/. (Accessed: 10th December 2018).

[3]. Microfluidic Solutions – High Performance Fluidic Flow Control – CorSolutions. Available at: https://www.mycorsolutions.com/fluid-delivery.html. (Accessed: 10th December 2018).

[4]. Brower K et al., An open-source, programmable pneumatic setup for operation and automated control of single- and multi-layer microfluidic devices, HardwareX 3 (2018) 117–134. [PubMed: 30221210]

[5]. Li B et al., A smartphone controlled handheld microfluidic liquid handling system, Lab Chip 14 (2014) 4085–4092. [PubMed: 25182078]

[6]. White JA, Streets AM, Controller for microfluidic large-scale integration, HardwareX 3 (2018) 135–145. [PubMed: 30775638]

[7]. Piraino F, Volpetti F, Watson C, Maerkl SJ, A digital-analog microfluidic platform for patient-centric multiplexed biomarker diagnostics of ultralow volume samples, ACS Nano 10 (2016) 1699–1710. [PubMed: 26741022]

[8]. OEM – EP – Miniature Pressure Controller. Available at: http://ph.parker.com/us/12051/en/oem-ep-miniature-pressure-controller. (Accessed: 11th December 2018).

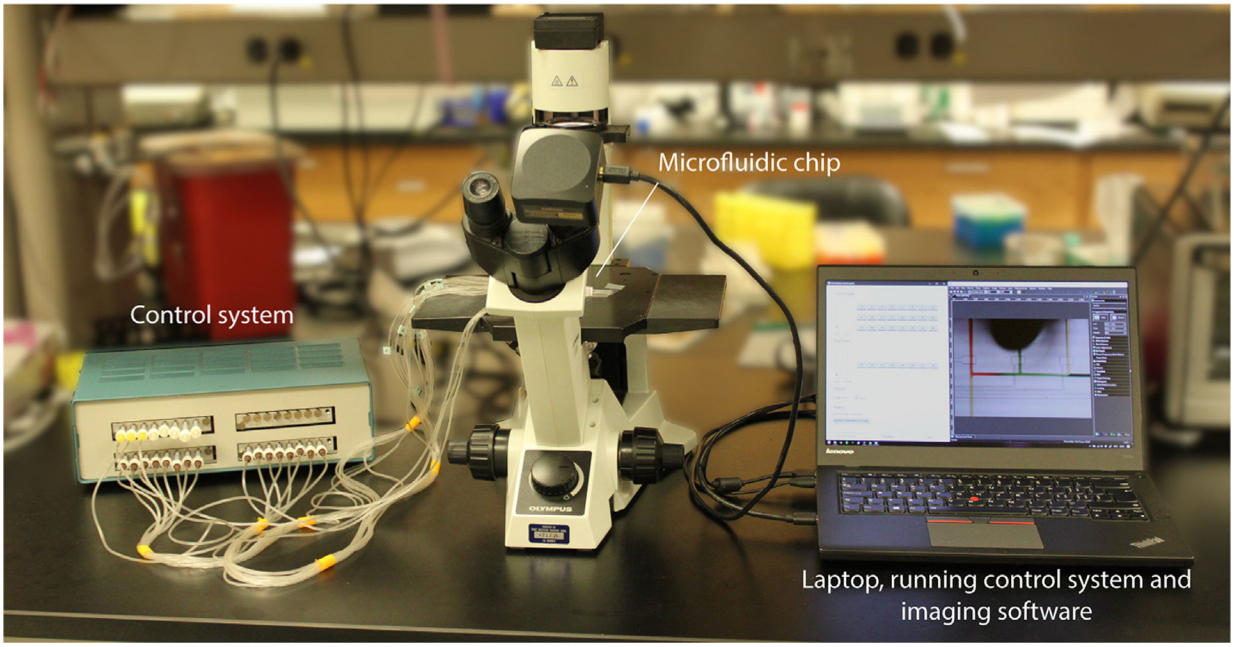

Fig. 1.

Photograph of the microfluidics control system (left) in use, connected to a microfluidic chip placed on the microscope (center). A laptop (right) runs the user interface for the system, as well as microscopy software to acquire images of the experiment.

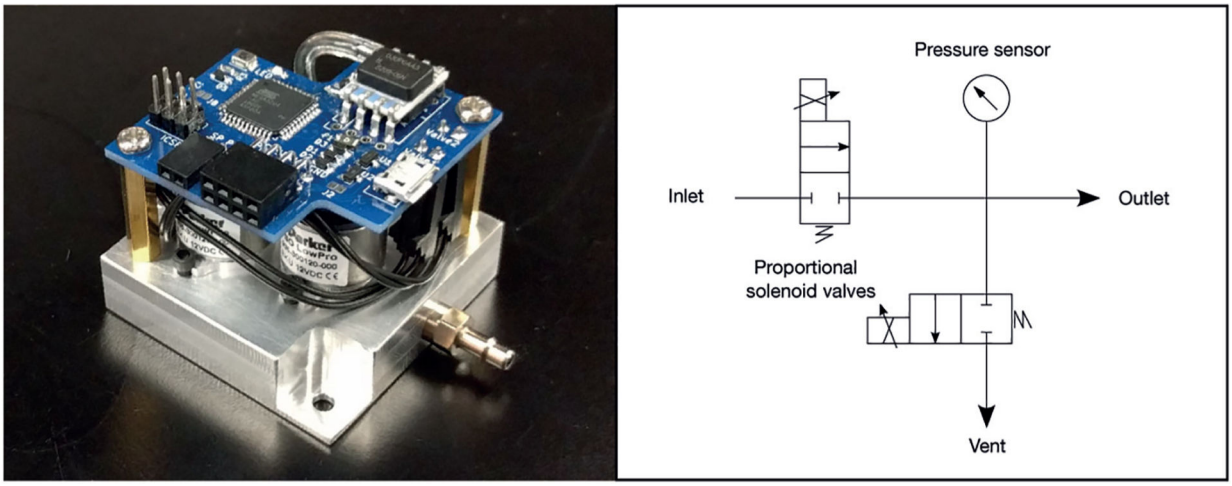

Fig. 2.

Photograph and schematic of the pressure regulator. A custom machined aluminum manifold provides ports for the inlet and outlet, two proportional solenoid valves, and a fitting to connect a pressure sensor. A PCB is mounted above the valves and includes a microcontroller, pressure sensor, valve drivers and connectors. The regulator measures approximately 4.5 cm × 4.5 cm × 4.5 cm.

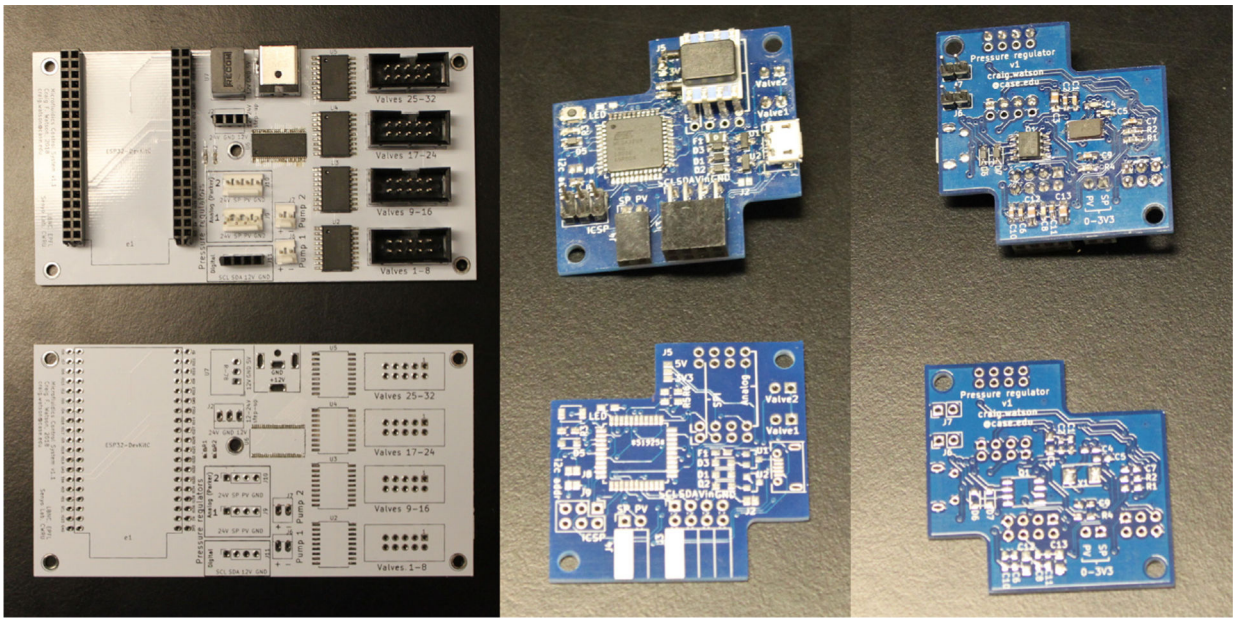

Fig. 3.

Main PCB (left) and pressure regulator PCB (right) before and after all components have been soldered on. Most components are mounted on the top side of the main PCB (back side not shown here), while the pressure regulator PCB has components on both sides to save space.

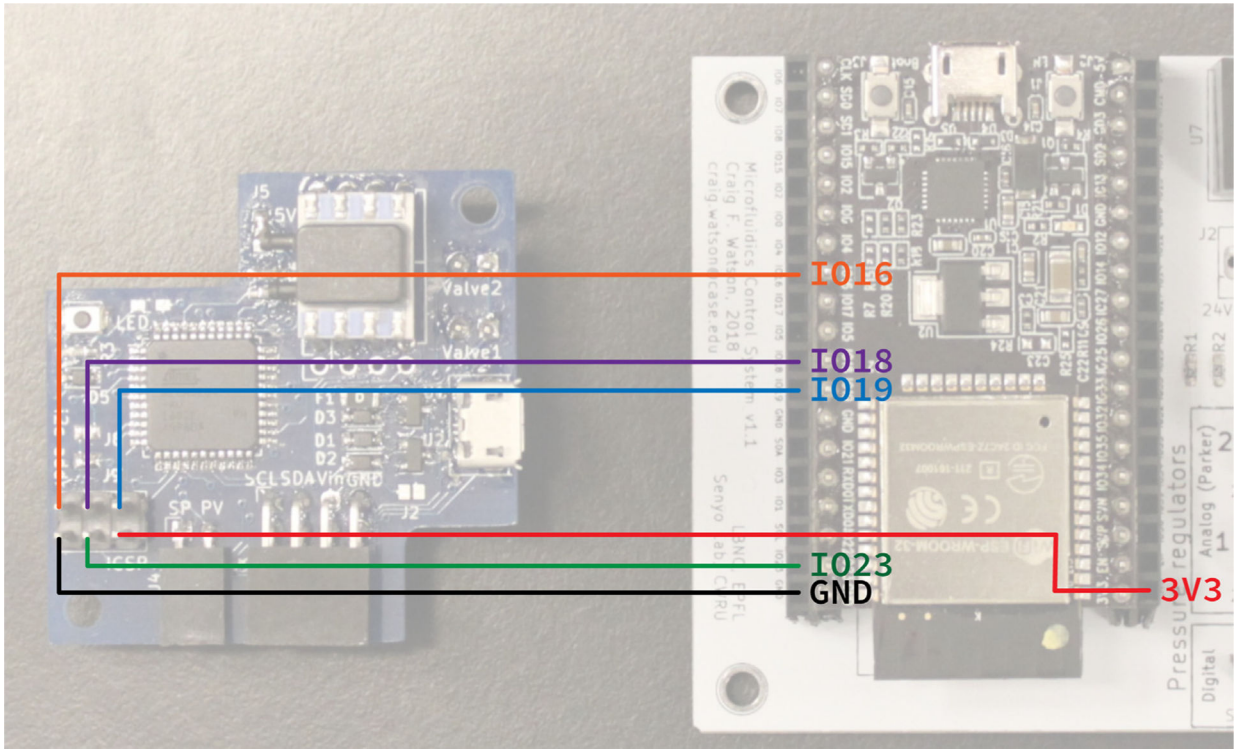

Fig. 4.

Wiring schematic to flash the bootloader onto the pressure regulator microcontroller. Labels indicate the pins on the ESP32. The ESP32 is then connected to a computer via USB (not shown), and the bootloader is flashed by the Arduino IDE.

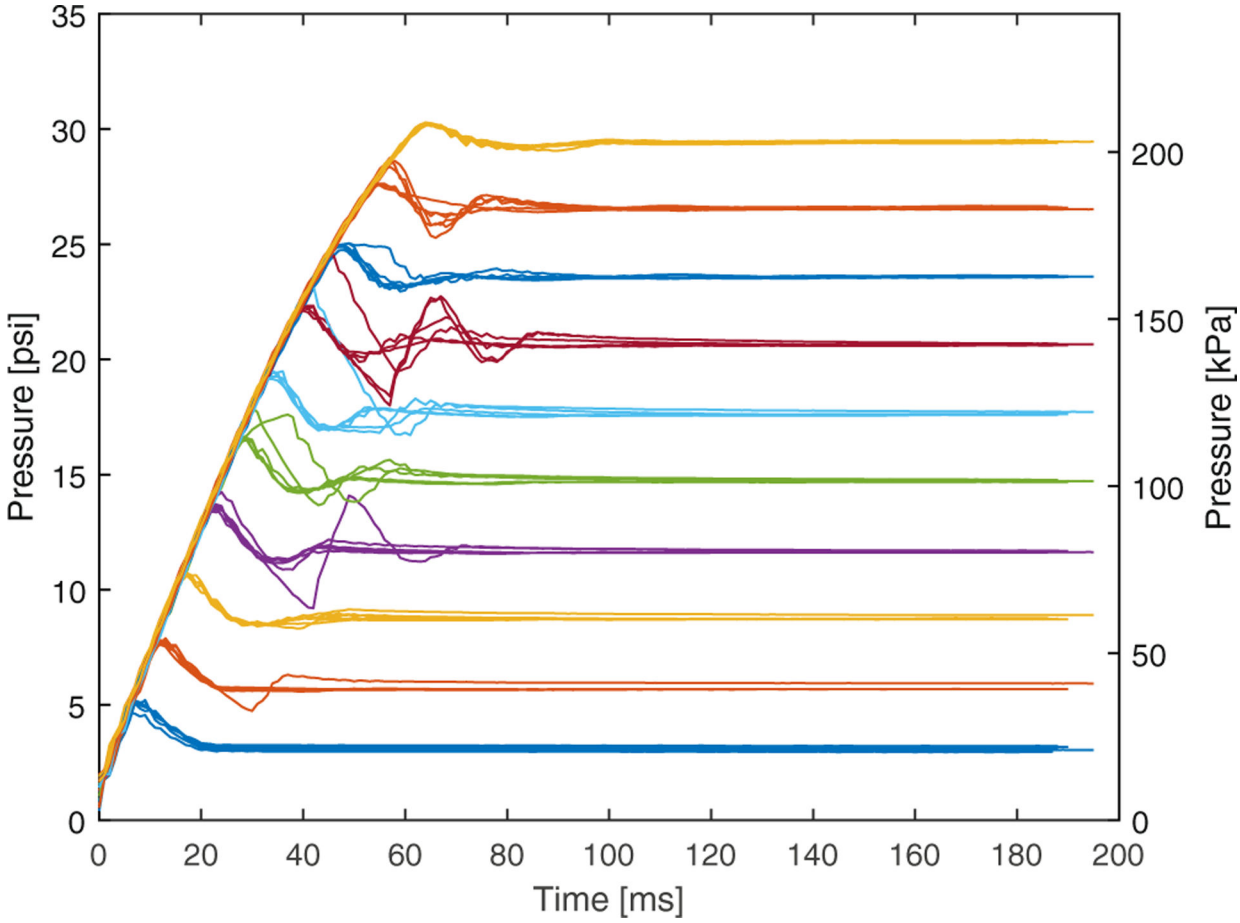

Fig. 5.

Step response of the 0–29.5 psi (0–203 kPa) pressure regulator. Setpoints of 10–100% of maximum range were sent over the i^2c interface by a second microcontroller. The resulting pressure was then measured using a sensor connected to the outlet of the pressure regulator. This was repeated a total of 6 times; all data is shown in this graph. Input pressure was 35 psi (241 kPa), supplied by a nitrogen tank with a two-stage pressure regulator. The initial delay was removed from this data, to align all trials on the same graph; the measured delay is reported in Table 1.

Table 1

characterization of the step response of the 0–29.5 psi (0–203 kPa) pressure regulator. Mean and standard deviation (SD) are based on 6 measurements, as in Fig. 5. The setpoint is transmitted over i^2c at t = 0. Delay is defined as the time for the measured pressure to exceed 5% of the setpoint. Settling time is defined as the time to settle within 5% of the setpoint; steady-state error is the absolute error at the final value measured.

Setpoint [psi]	Delay [ms]		Overshoot %		Rise time [ms]		Steady-state error %		Settling time [ms]	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
3.0	5.7	3.4	68.2	6.6	3.2	0.4	3.4	2.4	26.0	3.2
5.9	6.0	4.3	31.7	1.7	7.0	0.0	3.1	1.2	30.8	7.1
8.9	5.2	3.4	19.9	0.4	11.8	0.4	1.6	0.8	30.7	4.2
11.8	3.5	0.5	16.8	2.6	16.5	0.5	1.3	0.3	38.3	11.4
14.8	3.8	1.2	14.4	4.3	21.3	0.5	0.6	0.2	46.3	12.1
17.7	6.0	3.6	13.0	8.9	25.7	0.5	0.4	0.2	48.2	9.5
20.7	5.0	2.9	9.9	4.3	29.8	0.4	0.3	0.2	65.0	15.0
23.6	3.5	0.5	5.7	0.4	35.0	0.0	0.1	0.1	54.8	3.1
26.6	5.3	2.4	5.6	2.0	41.0	0.0	0.1	0.1	59.0	8.7
29.5	5.3	3.8	2.5	0.1	48.3	0.5	0.2	0.1	60.7	4.1

Table 2

Power consumption of the microfluidics control system, by component. Maximum values are based on parts' datasheets and validated through measurement, but do not reflect average power consumption, which depends on how frequently each component is powered on and is therefore application-specific.

Component	Maximum power, each [W]	Count	Max. power, total [W]
Pump	12	1	12
Pressure regulator	2.5	2	5
Solenoid valve	1.3	32	41.6
Microcontroller	<1	1	<1
		Total	59.6