LabSmith

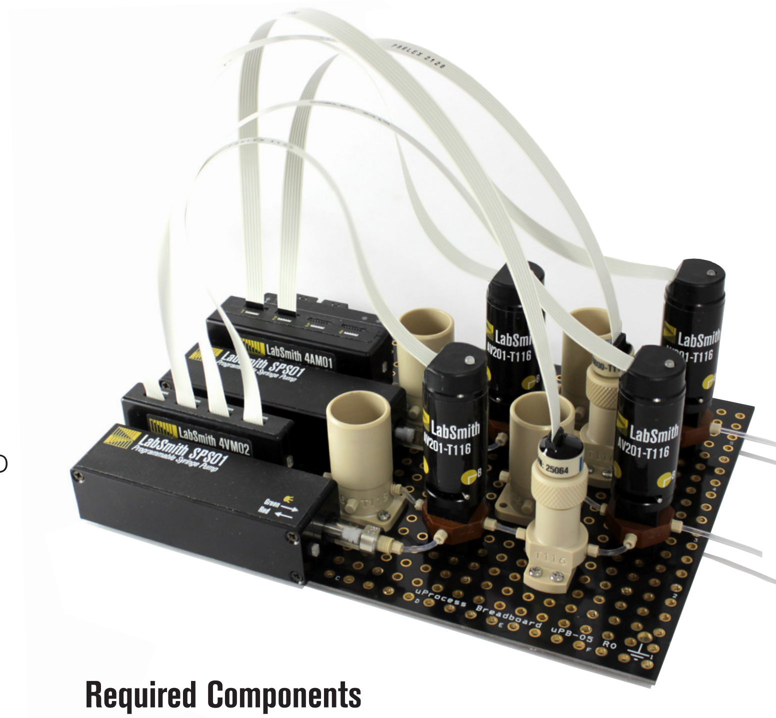
uProcess Flow Controller Automation

Introduction

uProcess flow controllers automate the maintenance of a regulation pressure and regulation flow rate over a set period of time. This app note primarily focuses on single-stream flow controllers, but [LabSmith's Flow Controller kit] includes everything needed to create and automate a two-stream flow controller.

There are two types of flow controllers currently supported. The first uses internal digital signaling to activate the syringe pump based on the sense pressure (FCA: Type A). The second controls the pump flow via a PID (proportional, integral, differential) controller algorithm based on the sense pressure (FCA PID: Type A PID).

Since no additional equipment is needed for a PID controller, it is recommended to use the FCA PID: Type A PID for more control and reduced fluctuations in data.



Contents

NOTE:
These instructions apply to uProcess Version 2.00.70 and later.

NOTE:
See www.labsmith.com/labsmith-application/uprocess-software-adding-a-new-interface for help adding new interfaces.

Required Components

LabSmith P/N	Description	Qty (1 Stream)	Qty (2 Streams)
uPB	uProcess Breadboard	1	1
EIB200	Electronic Interface Board	1	1
4VM02	Valve Manifold	1	1
AV201	Three-position, 3-port automated valve	1 (2*)	2 (4*)
4AM01	Four Channel Analog Manifold	1	1
uPS	Pressure sensor	1	2
SPS01	Syringe pump	1	2
BBRES-1ML	Reservoir w/ cap and O-ring (1.1 ml volume)	1	2
BBRES-5ML	Reservoir (5 ml volume)	1 (2*)	2 (4*)
[SIZE]-100	One-piece fitting, to connect capillary	7+	14+
[SIZE]-101	One-piece plug, to plug unused ports	3+	6+
BB-Tools	Breadboard hardware and installation tools	N/A	N/A
Tubing	at least 1 m tubing	1-2	2+

* optional

Fig. 1 - Components needed for flow controller(s)

Flow Controller Setup

1) Mount components to the breadboard, with a setup similar to that in **Fig. 2**.

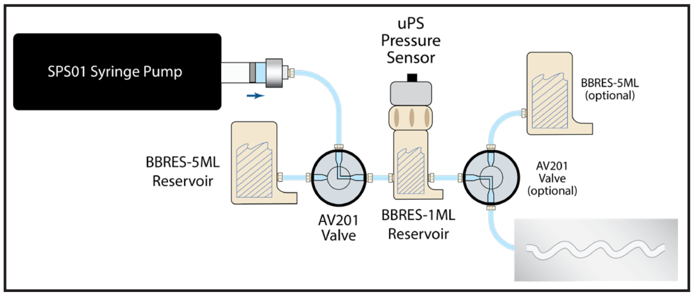


Fig. 2

2) In uProcess, click **Rescan for devices** in the toolbar, then ensure all devices appear under **Interfaces** in the **uDevices panel** (Fig. 3 & Fig. 4). If more than one valve, pressure sensor, or syringe pump is connected, give them each unique names. (Fig. 4).

Fig. 3

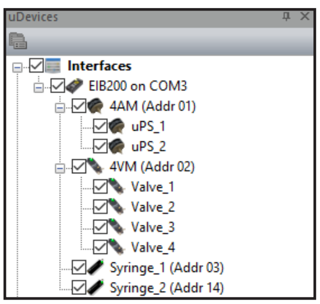


Fig. 4

3) Perform a leak check, and test connections to ensure all devices are communicating properly.

> **NOTE:**
> See www.labsmith.com/labsmith-application/how-to-make-leak-free-zero-dead-volume-microfluidic-connections for help performing a leak check.

4) In the **uDevices panel**, right-click **Assemblies**. Then hover over **New assembly** and select **Flow Controller** (Fig. 5).

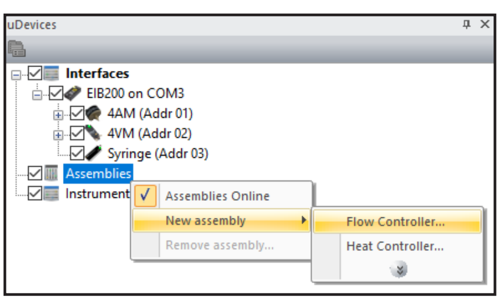


Fig. 5

5) Name the flow controller, and select the appropriate type from the dropdown menu (Fig. 6).

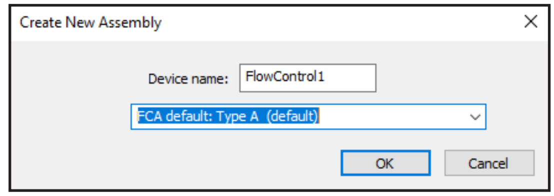


Fig. 6

6) The flow controller will now be listed under **Assemblies**.

> **NOTE:**
> The icon will have a red circle with a slash through it, to indicate that the flow controller is not online. This will be resolved after assigning parts.

7) Right-click on the flow controller and select **Assign parts...** (Fig. 7).

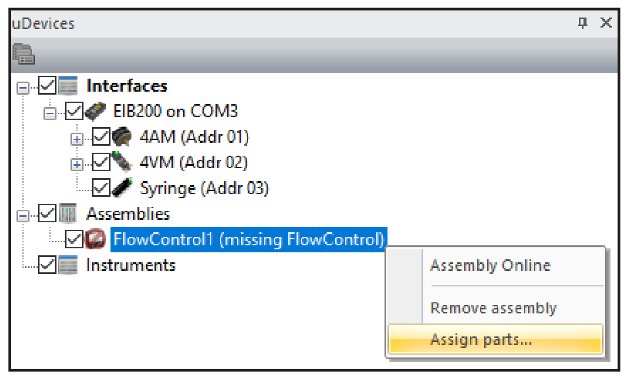


Fig. 7

8) Use the dropdown menu to select the proper pump, valve, and pressure sensor (Fig. 8).

> **NOTE:**
> If the correct valve does not appear, the valve type may be configured incorrectly. Return to the uDevices panel, and select your valve. Ensure the AV201 type is selected, then return to assigning parts to your flow controller.

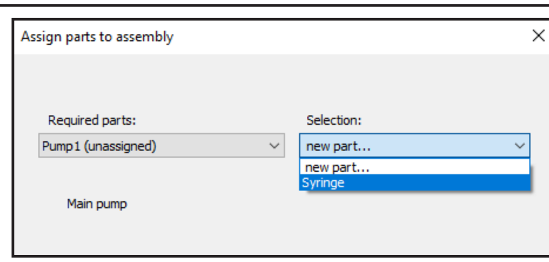


Fig. 8

9) For **Ch 1 (digital feedback channel)**, select one of the **EIB channels, A–D** (Fig. 9).

10) Click **OK** to finish assigning parts.

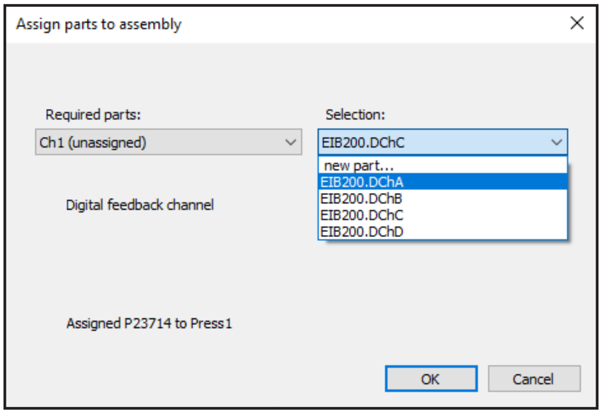


Fig. 9

11) In the **uDevices panel**, right-click on the flow controller again. This time, select **Assembly Online**. This will remove the red slashed circle (Fig. 10 & Fig. 11).

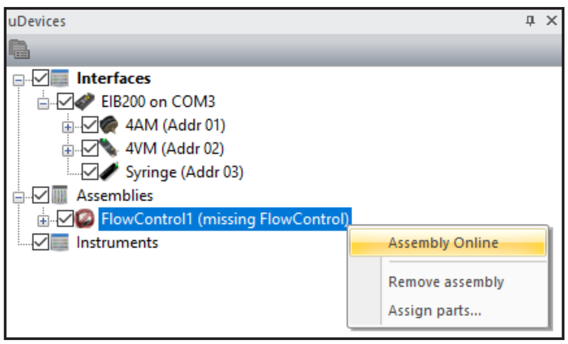


Fig. 10

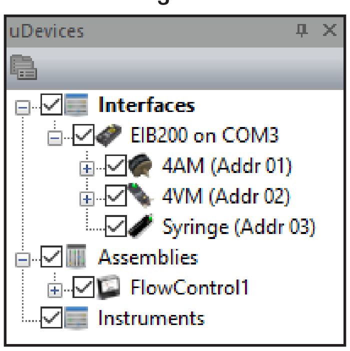


Fig. 11

> **NOTE:**
> For two streams, repeat Steps 4 – 11 to add a second flow controller. Ensure each flow controller has a unique name and uses a different EIB channel.

12) In the **uDevices panel**, click **Interfaces** and scroll down to the flow controller (Fig. 12).

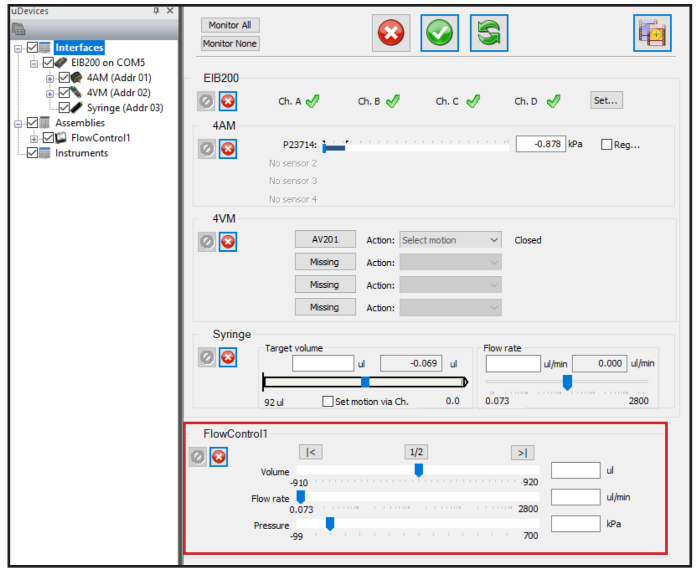


Fig. 12

Once your flow controller is set up, you can begin writing your script in the **Sequencer window** in uProcess. **Fig. 13** provides an example of the headers that are created in the **Sequencing Script** after a flow controller is added.

```
*EIB200.DChA    =    DCh
*P23714    =    uPS 800 kPa                1
*AV201    =    AV201
*Syringe    =    SPS 80 ul
*FlowControl1    =    FlowControl A
{
    Pump1 = Syringe ; Main pump
2   Valve1 = AV201 ; Input valve
    Press1 = P23714 ; Feedback pressure
3       sensor
    Ch1 = EIB200.DChA ; Digital feedback
        channel
4   CoarseRateFactor = 0.6 ; Fraction of max
        rate to use during coarse control
5   CoarseTol = 5 ; Pressure (kPa) variation
        tolerance during coarse control
6   DispensePos = 1 ; Valve position when
        dispensing
7   FillRateFactor = 0.6 ; Fraction of max
        rate at which to refill and re-empty
        syringe
8   FineTol = 0.01 ; Pressure (kPa)
        variation tolerance in fine control
9   PreloadDur = 0.2 ; Duration (s) to
        reverse pump before switching
        (backlash)
}
```

Fig. 13

1 **Headers** start with an * and should not be changed.

2 Inside the brackets are the flow controller **fixed parameters**. The first 4 lines are the assigned parameters and should not be changed. The last 6 lines are parameters which can be modified.

3 The regulation pressure and regulation flow rates are set in the script via **SetflowRate** and **SetPressure**, so they are not listed in the fixed parameters.

4 The **CoarseRateFactor** is a multiplier on the max flow rate. It sets the flow rate used during coarse pressure control. The SPS01 max flow rate depends on the syringe model.

> **NOTE:**
> If the flow rate is not set in the script, the coarse flow rate will be used for the fine control.

5 The flow controller has a pressurization control algorithm with two stages: **Coarse control** is used when the pressure is outside the **CoarseTol** (coarse tolerance) value, and **Fine control** is used when the pressure is within the **CoarseTol** range.

6 The value used for **DispensePos** (dispense position) will depend on the fluidic setup. For AV201 valves, Position 1 is Port A and Position 2 is Port B.

7 The **FillRateFactor** is the flow rate multiplier used for refilling the syringe.

8 When the pressure is within the **FineTol** value, the pressure regulation is satisfied and the SPS01 syringe pump will be idle until the pressure goes outside the **SetPressure** +/- **FineTol**.

9 **PreloadDur** (preloading duration) is the time in seconds it takes the SPS01 to dispense after filling but before switching the valve. This is to eliminate syringe backlash. This value is typically not modified.

Script Wizard

The easiest way to script in uProcess is to use the built-in Scripting Wizard. From there, the code can be modified as needed. See www.labsmith.com/labsmith-application/introduction-to-uprocess-and-using-the-script-wizard for more help.

1) In the **uDevices panel**, click **Interfaces** and scroll down to the flow controller.

2) Adjust the target volume, flow rate, and/or pressure, then scroll back up and click the **Save As New Sequencing Step** button (Fig. 14).

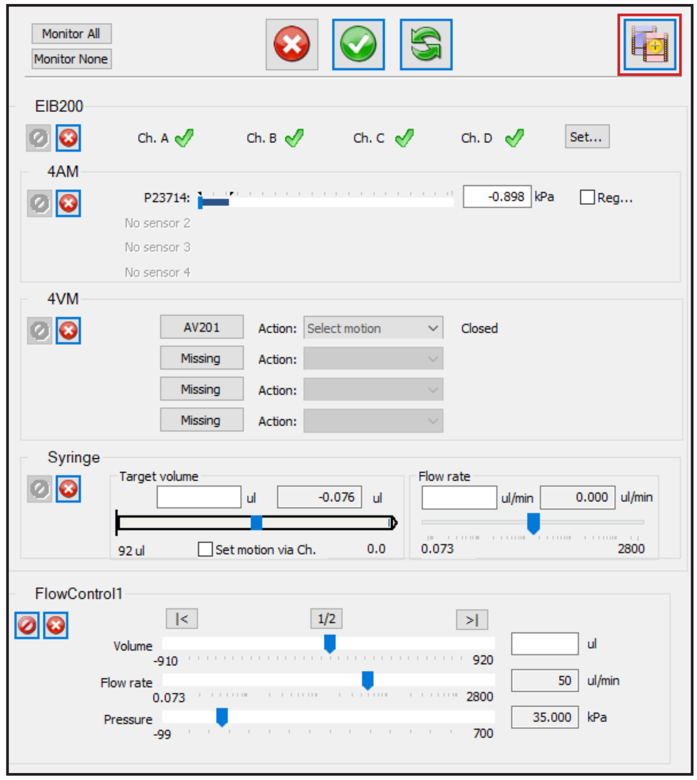


Fig. 14

3) Modify the step name and continuation settings if desired, then click **OK** (Fig. 15).

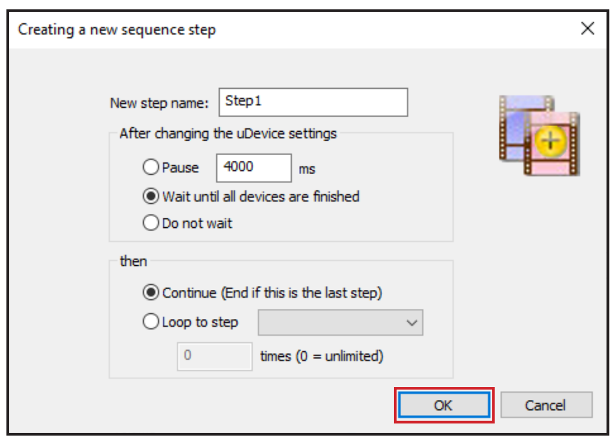


Fig. 15

> **NOTE:**
> The **Wait until all devices are finished** option should be selected, to ensure target values are reached before the script continues.
>
> To dispense for a specific time period, both **`WaitDone()`** and **`Wait(# s)`** can be used together, as shown in **Fig. 17**.

The code created using the **Script Wizard** can then be modified or expanded, either by adding additional steps via the **Wizard** or by manually typing in the step.

`Log(on)` Creates a log file. See www.labsmith.com/labsmith-application/logging-data-in-uprocess for help with logging data.
`Log("`*FILE PATH*`")` Creates a log file at the specified directory. The file name must be in quotes and end with .csv. If you include a directory, make sure that it exists and you have access. Ex. `Log("C:\Users\Lab\Documents\Logs\filename.csv")`
`Log(off)` Stops data collection
FLOWCONTROLLERNAME`: SetFlowRate(# ul/min)` Sets flow rate Ex. `FlowControl1: SetFlowRate(35 ul/min)`
FLOWCONTROLLERNAME`: SetPressure(# kPa)` Sets flow controller pressure Ex. `FlowControl1: SetPressure(20 kPa)`
FLOWCONTROLLERNAME`: WaitDone()` Advances script when pressure reaches set value Ex. `FlowControl1: WaitDone()`
`Wait(# s)` Sets an amount of time to wait before advancing script to next command Ex. `Wait(10 s)`
FLOWCONTROLLERNAME`: Stop()` Stops flow controller and moves valve to closed position Ex. `FlowControl1: Stop()`

Fig. 16 - Table of Relevant Methods for Sequencing Script

```
Step1:
   Log(on) ; creates a log file
   FlowControl1: SetFlowRate(50 ul/min)
      ; flow rate used during fine control,
      but, if not set, coarse rate is used
   FlowControl1: SetPressure(35 kPa) ; set
      flow controller pressure
   FlowControl1: WaitDone() ; advances when
      conditions are satisfied (pressure
      reaches set value)
   Wait(10 s) ; waits 10 seconds before
      moving to next command

Step2:
   FlowControl1: SetPressure(40 kPa)
   Wait(10 s)
   FlowControl1: Stop() ; stops flow
      controller, and valve moves to closed
      position
   Log(off) ; stops logging data to file
```

Fig. 17 - Sample Flow Controller Script

NOTE:
For the methods shown in **Fig. 16** and **Fig. 17**, the default units are shown, but other units may be used by specifying them in the script instead. If no units are added, the default will be used.

Maintaining Pressure

The following instructions will show how to run a flow controller until a target pressure is reached. See **Fig. 18** for the full script.

1) Use ***FLOWCONTROLLERNAME*****`: SetFlowRate(# ul/min)`** to set the flow rate and ***FLOWCONTROLLERNAME*****`: SetPressure(# kPa)`** to set the pressure.

2) Add ***FLOWCONTROLLERNAME*****`: WaitDone()`** to the script. This will ensure that the flow controller reaches the target pressure at the specified flow rate before continuing to the next command.

3) Add any additional commands/steps to your script, and the flow controller will maintain its pressure throughout.

TIP:
Be sure to add ***FLOWCONTROLLERNAME*****`: Stop()`** or the flow controller will continue to run, even once the script finishes.

```
Step1:
    Log(on) ; creates a log file
    FlowControl1: SetFlowRate(50 ul/min)
    FlowControl1: SetPressure(35 kPa)
    FlowControl1: WaitDone()
    FlowControl1: Stop()
```

Fig. 18

Dispensing a Set Volume of Fluid

The following instructions will show how to run a flow controller until a target volume of fluid is dispensed at pressure. See **Fig. 19** for the full script.

> **NOTE:**
> Although the basic flow controller will work with a single AV201 valve, a second can be added to prevent flow to your device while the pressure chamber is being primed, as shown in **Fig. 2**. Start with this valve in the closed position to help reach your target pressure.
>
> Since this valve will exist outside of the flow controller, the script controlling when it opens and closes will have to be added manually, as shown in **Fig. 19**.

1) Running a flow controller until a specified volume of fluid is dispensed requires scripting a conditional loop. uProcess automatically tracks dispensed fluid volume in a variable called **DispensedVol**, but this number will include the amount used while bringing the flow controller up to pressure.

2) Start by creating a variable for the target volume at the start of the script . For this example, this variable will be called **targetVolume**.

3) Use ***FLOWCONTROLLERNAME***`: SetFlowRate(# ul/min)` to set the flow rate, and use ***FLOWCONTROLLERNAME***`: SetPressure(# kPa)` to set the pressure.

4) Add `WaitDone()` to ensure that the flow controller has reached and is maintaining the specified pressure.

> **TIP:**
> If you are using the optional second AV201 downstream of the flow controller, open it here.

5) After waiting, use ***VARIABLENAME*** `=` ***FLOWCONTROLLERNAME***`.DispensedVol` to create a new variable to hold the value of the built-in **DispensedVol** at the moment that the target pressure is reached. For this example, this variable will be called **volAtPressure**.

6) The total dispensed fluid should be the sum of the volume of fluid dispensed while reaching pressure and the volume of fluid set to dispense at that pressure. Create a While loop with the condition `While (`***FLOWCONTROLLERNAME***`.DispensedVol <` ***STEP5VARIABLE*** `+` ***STEP2VARIABLE***`)`.

7) Inside the loop, add `Wait(`***TIME*** `s)` to give the script when to recheck if the condition is fulfilled. For this example, the script will wait for 1 second.

8) Outside the loop, once the target volume has been reached, be sure to add ***FLOWCONTROLLERNAME***`: Stop()`, or the flow controller will continue to run, even once the script finishes.

```
Step1:
   targetVolume = 200
   FlowControl1: SetFlowRate(50 ul/min)
   FlowControl1: SetPressure(35 kPa)
   FlowControl1: WaitDone()
   4VM: SetValves(0, 1, 0, 0) ; opens
      downstream valve to Position A
   volAtPressure = FlowControl1.DispensedVol

   While (FlowControl1.DispensedVol <
    volAtPressure + targetVolume)
   {
       Wait(1 s)
   }

   4VM: SetValves(0, 2, 0, 0) ; closes
      downstream valve
   FlowControl1: Stop()
```

Fig. 19

Optimization

While a flow controller ideally regulates flow pressure to a set point instantaneously and without fluctuations, this can be difficult to achieve due to occasional momentary changes, such as the syringe pump refilling or the valve opening and closing.

Flow Rate Adjustment	Pros	Cons
Higher	Reaches target pressure faster	Flow rate is less steady
Lower	Flow rate is steadier	Reaches target pressure slower

Fig. 20

Optimization will require experimenting with raising and lowering the flow rate until an ideal balance of reaching target pressure and maintaining a steady flow rate is achieved. It is suggested to use data logs to track results after each adjustment.

> **TIP:**
> The SPS01 set flow rate should be higher than the expected out-flow rate to maintain steady pressure.

A PID controller allows for even finer control and adjustment via three parameters, known as gain terms, used to regulate flow rate.

The three gain terms are: **proportion** (prop), **integral** (int), and **derivative** (deriv), or PID.

To adjust the PID gains, click on the flow controller in the **uDevices Panel**, under **Assemblies**. Then click the **Configuration...** button.

Fig. 21 can be used to adjust the PID gain values.

Issue	Prop.	Int.	Deriv.
Overshooting: pressure exceeds target value before settling	increase	increase	decrease
Extended Rise Time: target pressure takes too long to reach	decrease	decrease	minor increase/ no change
Extended Settling Time: takes too long to settle to consistent target pressure	minor increase/ no change	increase	decrease

Fig. 21

> **NOTE:**
> Adjust the PID gains in real-time, to see the changes that follow each adjustment. Alternatively, data logging can be used to track results after running the script.

The **Configuration window** has two additional variables which can be adjusted: **Course Tolerance** and **Fine Tolerance**.

Course Tolerance is the pressure (kPa) variation tolerance and is used during coarse control.

Fine Tolerance is the temperature (C) tolerance and is used during fine control.

After adjusting variables, click **OK**.

Verifying Results

The pressure and flow rate for the setup should be calculated empirically and used as a comparison, to ensure the flow controller is producing the expected results.

Fig. 22 shows the equations used, where **ΔP** is the change in pressure, **R** is the resistance, and **Q** is the mean flow rate. Microfluidic resistance calculators can also be found online.

$$\Delta P = R * Q$$

For rectangular channels

$$R = \frac{1}{1 - 0.6\,(h/w)} \, \frac{12\,\mu L}{h^3 w}$$

For round tubing

$$R = \frac{128\,\mu L}{\pi d^2}$$

Fig. 22

Common Issues

Start by updating your device firmware before attempting further troubleshooting:

1) Ensure that uProcess is up to date by clicking **Help** in the top menu bar, then clicking **About uProcess...** Check that you are using Version 2.00.70 or later.

2) Now check the firmware of your devices by connecting them and rescanning. Be sure they all appear in the **uDevices panel**.

3) Go to the **Actions** menu, select **Firmware**, and click **Update All Firmware**.

Issue 1: Valve isn't appearing when defining assembly parts

1) Be sure that the Valve type is correct. For FCA and FCA PID controllers, an AV201 is needed.

> **NOTE:**
> You can use an AV202 valve instead (with a plug in the 4th port), but it must still be declared as an AV201 for the flow controller setup to recognize it.

2) In the **uDevices panel**, click on the valve under **Interfaces**.

3) Click the name of the valve to open the **Valve configuration window**. This is *not* the same as clicking the **Configuration...** button.

4) Click **OK**.

5) The valve should now appear as an option in the dropdown menu when defining assembly parts.

Issue 2: Pressure/Flow rate are higher/lower than expected

1) If the **pressure is higher** or the **flow rate is lower** than expected, check your set up for clogs or kinked tubing. You can also try decreasing the back-pressure.

2) If the **pressure is lower** or the **flow rate is higher** than expected, check your set up for leaks. You can also try increasing the back-pressure, for instance, with a long length of tubing on the out-flow.

Issue 3: Pressure/Flow rate drop while syringe refills

1) Consider using a dual-syringe setup for continuous flow.

2) See https://labsmith.com/labsmith-application/creating-pressure-driven-flow-with-labsmith-udevices/ for more information.

Issue 4: Syringe can't reach 0 volume

1) In the **uDevices panel**, under **Interfaces**, click on the syringe pump.

2) Click the **Configuration...** button.

3) In the **Calibration** section, click **Calibrate**.

4) If issues persist after calibration, go back to the **Calibration** section and adjust the **Full Out** value. A good starting point is adding 200 to the current number.