User Manual

Dylan Patterson
Jonna Jämte
Karl Asklund
Lucas Sevelin
Martin Ling
Theodor Vallgren

December 6, 2023

Version 1.0

Status

Reviewed	Dylan Patterson	2023-11-29
Approved	Lars Eriksson	2023-12-04

Project Identity

Group E-mail: tsrt10-evap@groups.liu.se

Homepage: https://tsrt10.gitlab-pages.liu.se/2023/aurobay

Orderer: Lars Eriksson, Linköpings universitet
Phone: +46 (0)13-28 44 09
E-mail: lars.eriksson@liu.se

Customer: Fredrik Wemmert, Aurobay
Phone: N/A
E-mail: fredrik.wemmert@aurobay.com

Supervisor: Oskar Lind Jonsson
Phone: N/A
E-mail: oskar.lind.jonsson@liu.se

Course Responsible: Daniel Axehill
Phone: +46 (0)13-28 40 42
E-mail: daniel.axehill@liu.se

Participants of the group

Name	**Responsible**	**E-mail**
Dylan Patterson	Responsible for the quality (QS)	dylpa851@student.liu.se
Jonna Jämte	Responsible for the testing (TST)	jonja121@student.liu.se
Karl Asklund	Responsible for the Documentation (DOC)	karas744@student.liu.se
Lucas Sevelin	Responsible for software (SWR)	lucse807@student.liu.se
Martin Ling	Project leader (PL)	marli498@student.liu.se
Theodor Vallgren	Responsible for the design (DES)	theva365@student.liu.se

CONTENTS

DOCUMENT HISTORY

Version	Date	Changes made	Sign	Reviewer
1.0	2023-11-29	First draft.	Dylan Patterson	Martin Ling

1 INTRODUCTION

This document is the user manual for the project "Modeling and Control of the Purge System" in the course "Project course in Automatic Control, CDIO, TSRT10" at LiU. This document is used as a guide to operate the measuring system and test bench for use in this project.

2 HARDWARE

The main hardware used in this project is the purge test bench, the measuring system, and the carbon canisters. Complementary hoses and similar connections are also used to connect these components.

2.1 Purge Test Bench, provided by Aurobay

The provided test bench consists of a controlling computer and a gray box with internal connections. It also has out- and inputs for the connection to the carbon canister and to the provided vacuum pump on the rig.

Internally two flow sensors are installed (excluding the measuring system's own flow sensor) and one also acts as a valve. The valve is controlled by the system provided by Aurobay and the data given by the flow sensor is the flow out of the system.

2.2 Measuring system, built by the project group

The measuring system consists of a large wooden board where the temperature sensors and the controlling system for these are mounted as shown in fig. 1.

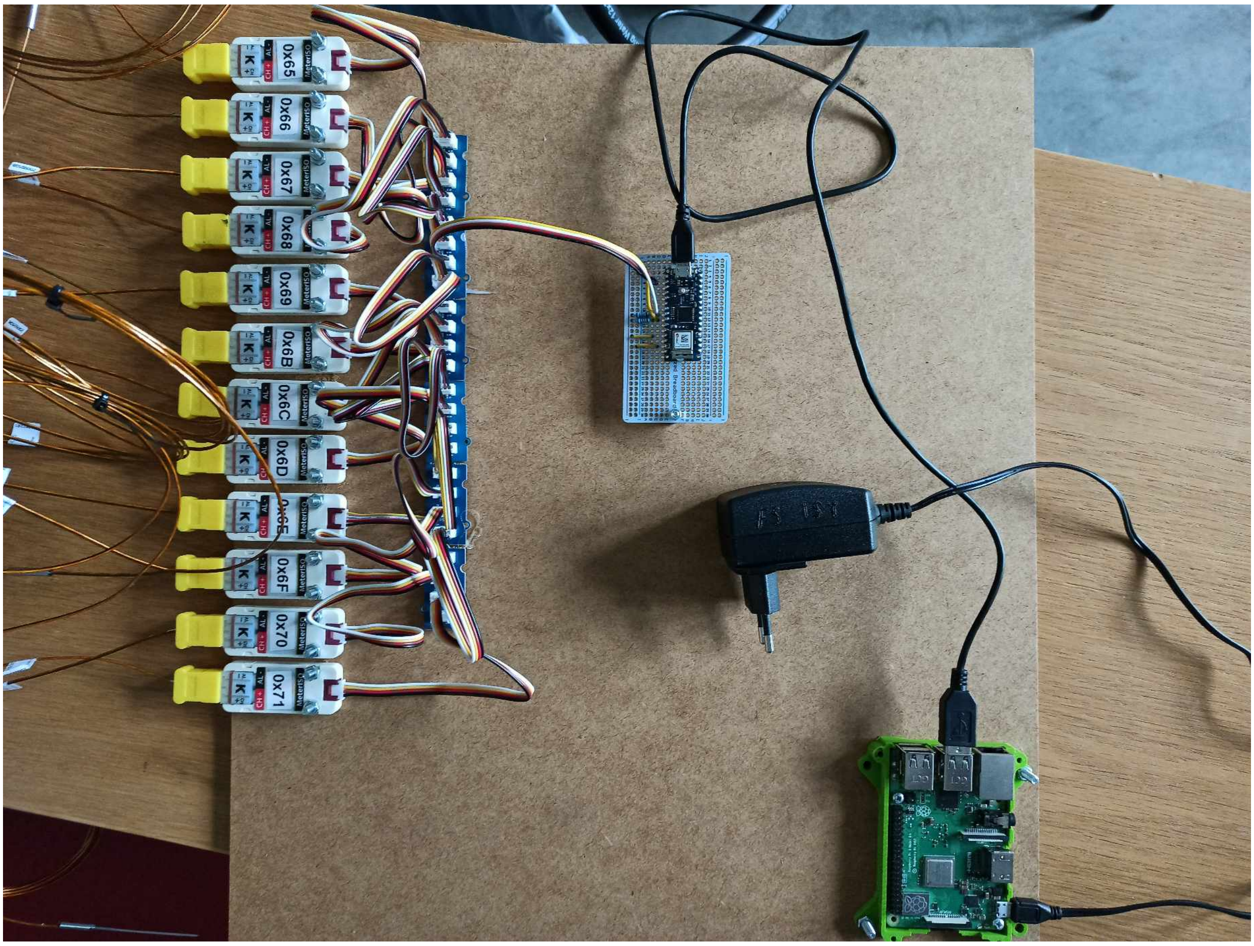

Figure 1: Measuring system setup

In fig. 2 the connections on the Arduino pins and sensor connections are shown.

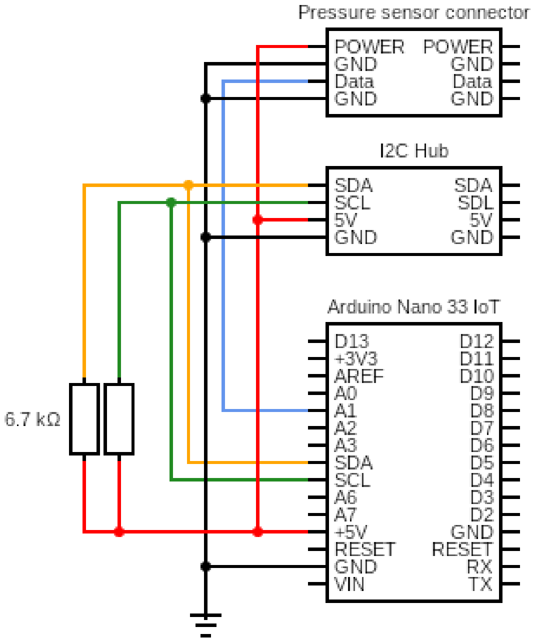

Figure 2: Wiring diagram

3 SOFTWARE AND USAGE

3.1 Running the data logging software

1. Connect the Arduino to the Raspberry Pi via USB, the USB should be connected to the top-left port.
2. Connect the Raspberry Pi to power.
3. Connect to the Wi-Fi on the Raspberry Pi. The Raspberry Pi can sometimes take a few minutes to start.
   - SSID: Datalog
   - Password: `Aurobay2023`
4. Connect to the Raspberry Pi over the SSH protocol (Remember to reset the Arduino before every measurement)
   - **On Windows**: Use for example Putty with the address `10.42.0.1` and the username `aurobay`
   - **On Mac/Linux**: Use the command `ssh aurobay@10.42.0.1`
   - The password is `kaffe`

5. Run the Python script using the command `python "./Desktop/Aurobay_2023/log_data.py"`
6. Input the number of seconds to run the logger, it is recommended to do a few more seconds than the actual cycle.
7. When the program has completed its measurements it will output the total time measured as well as a file name. The file name is similar to `log_XXX.csv` where `XXX` is a random number between 0 and 999.

Note! Because of the random file naming there is a possibility that the new file will overwrite an old file, it is therefore recommended to always move the files to your computer directly after completing the measurements.

3.2 Moving the file to your computer

To move the file to your computer using Wi-Fi, the following command can be used:

- **On Windows**: Use for example PSCP, and follow the same instructions as for Mac/Linux.
- **On Mac/Linux**: Open a new terminal and run

  ```
  scp aurobay@10.42.0.1:./Desktop/Aurobay_2023/Data/log_XXX.csv ./Desktop
  ```

  where you replace `log_XXX.csv` with the actual filename. This will place the file on your Desktop, if you want to place it elsewhere replace `./Desktop` with the path. On Mac, you can for example right-click in the folder you wish to place the file in and select ”Copy location as Pathname” to get the absolute path of the folder.

Another alternative is to plug a USB drive into the Raspberry Pi, and using the the command `mv Desktop/Aurobay_2023/Data/log_XXX.csv "to-path"` the file can be moved to the USB-drive.

Finally, the Raspberry Pi has a GUI so by connecting the Raspberry Pi to a mouse, keyboard, and monitor it is also possible to move the file to a USB drive the same way you would do on any computer.

3.3 Gray-Aurobay Box usage

1. Check that all connections inside and outside the gray box are securely connected. Start the computer on the test bench.
2. Check all connections to and from the box so no gases can leak from the couplings. Also, check that the exhaust pipe is in a well-ventilated area or preferably in a hood ventilation apparatus (when hydrocarbons are charged in the canister). Along with this also check that all external sensors are mounted correctly and do not leak.
3. Start the test program that controls the flow to and from the canister and first open the **"Curve Creator"**.
4. When the curve creator is open, start an Excel document with the stored data for the test you want to conduct. Any number of elements in a column can be copied and pasted into the corresponding green box.
5. In the curve creator the tolerance can be adjusted if needed and the name for the test can be set. After all wanted information is written, click the generate button to finalize the curve.
6. When the curve is done, navigate back to the start page of the program and click **"Load Custom Trace"**. If everything is done correctly this will load the previously made curve. Also adjust the option for set volume in liters under the options tab if needed.

7. Now click the **"Log data"** button to start logging in order to save what the test bench is doing during the test.
8. Start the cycle by pressing the **"START"**-button and wait until the test is done.
9. After the test has ended the logged data can be stored as an Excel document.

3.4 Logging using the scale

To collect good data on the mass of the canister during the purging a Miller Toledo MS32001L is used, which can log data to a computer using the built-in USB-port. This is done as follows:

1. Start the scale
2. Open the menu, using the second button from the left.
3. Press the minus twice to navigate to the alternative `Interface`, press enter.
4. Press minus once, to get the option `Interval`, press enter.
5. Press plus once to increment the interval to 1, press enter.
6. Press the menu key, and once the prompt `Save? Yes` appears, press enter.

To read the data you need a Serial monitor, on MacOS you can for example use `SerialTools`, and on Windows, you can use the Serial monitor. The data should now be transmitted to the computer, remember to save the results.

3.5 Data Processing

In order to process the data MATLAB scripts was used to visualize the information in graphs and filter it in different ways for analysis. The model is also implemented in MATLAB. There are two options available to run the model, either by doing one optimization one the entire (or part of) the cycle or by running the model in specified intervals. These options are further described below. Also included in the repository is the file (`reportPlots.m`) used to make all plots in the results, this is however not further explained as that is not part of the results.

3.5.1 *Running with one optimization*

Open MATLAB, the version tested is 2021b. The MATLAB-file to run is: `runMe.m`. Make sure that the folder structure is the same as in GitLab in order for the script to run properly. The following variables can be altered:

1. `measurements` denotes which measurement to fetch data from. The available options are shown in table 1 in the column ”Model available”. Not all measurements are available as there must be data from the scale, at least one temperature and at least one flow measurement available.
2. `start_index` is the measurement index the model should start at, set to 1 if you want the entire cycle.
3. `end_index` is the measurement index the model should end at, set to 0 if you want the entire cycle.
4. `temp_sensor` is the index of the temperature sensor to use, set to ’auto’ to automatically select the best sensor.
5. `opts` allows to force certain values for k_1 and/or k_2 in the model (structure: opts = $[k_1, k_2]$), leave empty to run on automatic mode. To force value for k_2 but run k_1 on auto, set the value of k_1 to `inf`. To only force value of k_1 either set k_2 to `inf` or remove the value from the vector (making it a vector with only one value).

3.5.2 *Running with intervals of optimization*

Open MATLAB, the version tested is 2021b. The MATLAB-file to run is: `runMe_Intervals.m`. Make sure that the folder structure is the same as in GitLab in order for the script to run properly. The following variables can be altered:

1. `measurements` denotes which measurement to fetch data from. The available options are shown in table 1 in the column ”Model available”. Not all measurements are available as there must be data from the scale, at least one temperature and at least one flow measurement available.

2. `start_index` is the measurement index the model should start at, set to 1 if you want the entire cycle.

3. `end_index` is a vector of indexes to end the measurements at. The intervals can be of varied sizes, the last value must be a zero.

4. `temp_sensor` is the index of the temperature sensor to use, set to ’auto’ to automatically select the best sensor.

5. `opts` allows to force certain values for k_1 and/or k_2 in the model (structure: opts = $[k_1, k_2]$), leave empty to run on automatic mode. To force value for k_2 but run k_1 on auto, set the value of k_1 to `inf`. To only force value of k_1 either set k_2 to `inf` or remove the value from the vector (making it a vector with only one value).

3.6 Available sensors per canister

Table 1: Available sensor data for each canister.

Canister name	Scale	Temperature	Number of temperature sensors	Flow	Flow Greybox	Pressure	Model available
LIU1	No	Yes	8	Yes	No	No	No
LIU2	Yes	No	-	No	No	No	No
LIU3	Yes	Yes	12	Yes	Yes	No	Yes
LIU4	Yes	Yes	8	Yes	Yes	No	Yes
LIU5	Yes	Yes	8	Yes	Yes	No	Yes
LIU6	Yes	No*	9	Yes	Yes	Yes	No
LIU7	Yes	No	-	No	Yes	No	No
LIU8	Yes	Yes	2	No	Yes	No	Yes
LIU20	Yes	Yes	8	No	Yes	No	Yes

* = *Data is available, but only for the first 80 seconds*