**Introduction**

Welcome!

If you are reading this then you have the outstanding opportunity of using Bronkhorst Flow Microcontrollers, but you unfortunately put your trust in an Undergraduate programmer to do your flow automation (not the best choice!); however, this program is robust and correct enough that you can rest assured that your reactions will proceed as planned.

The software detailed throughout the rest of this manual is a python-based program that automatically monitors the current flows of multiple Bronkhorst controllers and changes the flows based on times specified by the user. The software takes the responsibility of monitoring the timing of flow changes away from the user and also offers protection to the experimenter in terms of ensuring the slows switch to the correct values. One of the biggest differences between this project and similar programs on github, is that this software allows for multiple microcontrollers to be used in parallel along with multiple computer ports to be used. Additionally, automated email updates to the user can be added to indicate the completion of the reaction or if there was an error. The interface is through the command line, and can be run on virtually any computer that can support python.

Throughout this manual general installation/configuration guidelines, usage procedures, and troubleshooting advice will be explained. Please note that this is not a substitute for the Bronkhorst Microcontroller manual. Please read both so that you understand how the controllers are setup, too. This is meant to be thorough, although I am sure that I will miss parts. Please update this manual on my github page if you find anything that could be explained better or if you think the addition of an explanation/procedure is needed. This is meant to be an open source project that anyone can and should contribute to if they would like to use Bronkhorst Flow controllers in their own research efforts.

The Autonomous Flow Monitoring System (AKA. AFMS) is originally a project for the Sustainable Chemistry and Catalysis Engineering (Ive Herman’s lab) at the University of Wisconsin-Madison. My main research mentor was Sarah Specht, who has been a great asset throughout the creation of this program as a willing experimenter for my software and as a patient “client”. My hope is that this software can be expanded on and used in other research labs throughout the country.

I wish you the best in your endeavor to use my software, and please do not hesitate to reach out to me on github, linkedin, or wherever you can find me. I hope this software helps!

Sincerely,

Ethan Young

**Notable Features**

*Automated Mass Flow Changes*

The main focus of this software project was to create an easy-to-use software system to control Bronkhorst Mass Flow Controllers. This software allows for the operation of these controllers in an arbitrary length series, and even allows for control across multiple ports.

For example, the researcher that inspired this software uses it to run 8 hour, sometimes overnight, reactions with flow changes every 21 minutes.

*Email Notifications*

In the UserComm.txt file, it is possible to specify various emails that can be notified about important events during your reaction. At the commencement of the program, it prompts the user to pick an email from the list specified in the file. The program will then send a notification to the specified email when either the reaction successfully completed or there was an error in the reaction.

*Emergency Flows*

Because of the dangerous nature of some reactions, it is important that if anything goes wrong the program is able to detect it and correct for it before the situation becomes more dangerous. The two main cases that the program accounts for in terms of error handling of flows is if a flow does not reach the specified flow or over shoots the expected flow. Both can be dangerous.

Some possible causes could be a flow that is above the maximum of the controller, a controller malfunction, or a canister of gas runs low. Regardless of the cause, it is useful to be able to have a set of flow rates that will be switched to in case of a reaction failure.

*Logging*

Accurate logs with Timestamps are important in chemistry research, specifically for process validation and for cross referencing to other systems in use. For example, the research for which this software was developed used a GC-MS afterwards, so it was helpful to be able to know the time that the flows changed to cross reference with the GC.

*Multiple Reaction Files*

Allowing for multiple reaction files in the Reactions folder helps in creating a more accurate and efficient process for reactions for a researcher. Being able to create reaction files ahead of time saves time for the researcher and increase the overall usability of the system.

**Installation and Configuration**

This section describes how to download the software package to your local computer and how to configure the different settings files for use of the program.

***Installing From Github***

1. Go to the github page with the software: <https://github.com/Ethan4242/Autonomous-Flow-Monitoring-System> or search for the Autonomous Flow Monitoring System on github
2. Download the files to your local computer
   1. Click the green “Clone or Download” button and do one of the options below..
      1. Download the zip file and extract it into the desired place on your computer (This manual will assume you extracted it to your desktop)
      2. Git clone the files onto your computer by navigating to the folder you would you like to save the files in terminal (or command prompt for Windows), then copy the link in the clone or download box, then type in terminal (for example)..
         1. git clone https://github.com/Ethan4242/Autonomous-Flow-Monitoring-System.git

***Confirming Correct Python Version***

This project can be used with any variant of python 3.x. To ensure that python is installed correctly and is the right version..

1. Type “python -V” into terminal or command prompt
2. If the computer says that it does not recognize the command “python”, this means that python is either not installed correctly or is not installed at all. I would recommend searching the web for how to install the newest 3.x version of python.
   1. For Windows users, the main trick when installing python is ensuring that the correct PATH to python is specified for the OS.
3. If the computer outputs a Python version but it is 2.x, then this means you have the older version of python, which does not work with this program. I would recommend searching the web for a way of updating to the newer version.

***Configuring your AFMS System***

This section will describe what should be put into the Settings Folder of the software package.

*Filenames.txt*

This file is used as the initialization file for the other files in the program. Its main purpose is to give more flexibility for another developer to change how files are accessed, how many are accessed, and where they are stored later on. If you would like to have the other files saved in a different location, then all you need to do is change the relative (to the python program) file path of each in this file. For example the defaults are as listed below..

*1) Reaction Inputs*

*./Reactions/ReactionInput.csv*

*2) Bronkhorst Settings*

*./Settings/BronkhorstConfig.txt*

*3) User Communication*

*./Settings/UserComm.txt*

If you do not wish to move around the file storage structure of this software, then leave this file as-is.

*BronkhorstConfig.txt*

This is an important file for this software package! This file contains the node address of each microcontroller, the maximum flow for each controller, and the user specified name for each controller. Please note: Any controller you indicate here, also needs to be added to the Reactions file, but that will be for a later section.

Initialize each controller as specified below..

*Name 1: CO2*

*Port 1: COM3*

*Node 1: 10*

*Max Flow 1: 100*

*Name 2: O2   
 Port 2: COM6*

*Node 2: 20*

*Max Flow 2: 30*

*……….*

There can be an arbitrary number of controllers that are spread over an arbitrary number of communication ports

* *Name:* This is the user defined name and will be used to refer to this controller for remainder of the program.
* *Port:* This is the communication port that the Controller is connected to. This can be the same across multiple controllers.
* *Node:* This is the port number specified by the Bronkhorst Controller system. It is in decimal.
* *Max Flow:* This is also a value specified by the unique microcontroller. The system will not explicitly ensure that the flow indicated by the user does not exceed this value, but there will be an error in the execution because the system will not be able to achieve this Max Flow.

*Settings/UserComm.txt*

This file holds the emails that are options for the email notification feature. The program will prompt the user to specify which email to notify from the list in this file. The number of emails can be arbitrary, although only one can be notified on a given reaction.

**Software Usage**

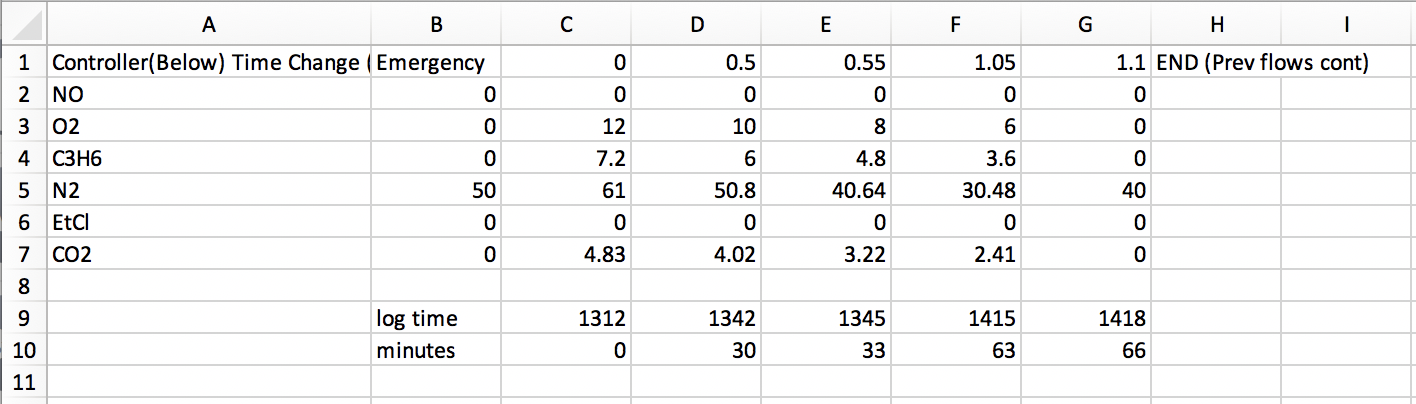
This section describes how to format your Reaction file to be run by the software and then actually run the program. There can be an arbitrary number of reaction files, and the user is prompted to specify which one they would like to use once the program begins.

**Saving the File**

The file must be saved in the location specified in the filenames.txt file. The rest of this manual will assume that the file is saved in the default location: ./Reactions.

**Formatting The Reaction Input File**

Below is a sample Reaction that is ready to be run. Please note that the format needs to be identical to this format for the program to be able to read the file. ***Additionally, make sure to save the file as a .csv file.***

**

*Headings*

The headings do not need to be the same as specified above, although the last cell in the first row (F1) needs to have END in it somewhere. This allows the program to know whether you are specifying another time interval or ending the program.

*Controller Names*

The controller names allow the program to determine which microcontroller you are referring to with a simple name. ***The controller names in column A need to be the same as those specified in the BronkhorstConfig.txt file, and ALL of the names need to be specified.*** If a name is specified in the Config file and not the ReactionInput file, then the program will throw an error (and vice versa). I would recommend to keep all of the controller names in the ReactionFile and just set their flows to 0 when you are not using them.

*Emergency Flow Controls*

The emergency flows are the flows that the controllers switch to if there is a major error during the reaction, such as if there is not enough gas in one of the tanks to continue the reaction. The flows will be switched to the emergency flows, an email will be sent, and the program will cease to run. The second column always needs to remain the emergency flows column.

*Time Change Flows*

The main flows are the flows each controller is set to until the next flow change is specified. In the first row, between the emergency flow column and the END column, are the time changes for the flows. Below these times are the corresponding flows for each controller at that specific time.

The times in the first row is the time elapsed since the reaction started in fractions of an hour. Thus 0.0333 is 2 minutes and 2.5 is 2 hours and 30 minutes. Please note that *you must specify the flows* *at 0* elapsed times, which are the initial reaction flows. Also note that the last time you specify are the ending flows. These ending flows will continue indefinitely until the user starts a new reaction or shuts them off manually.

The flows below a given elapsed time are the flows the controllers will switch to after that amount of time has elapsed since the reaction started. ***Each controller needs a flow during every specified time, even if it is just 0.*** If a flow is not specified, then the program will not be able to read the file.

**Running a Reaction**

*Executing Program*

At the point of writing this manual, the only way of running this program is via the command line. Thus to execute the program, you will need to navigate to the folder containing Reaction.py.

For example, the sequence might be for a Windows user..

* “Dir”
* “Cd Desktop”
* “Dir”
* “Cd FlowSoftware”

To Execute the software, you need to type: “python Reaction.py”. Then the software will begin and start prompting the user for various settings that it will need to begin the reaction.