Virtual Moon Project: Documentation

# *Project Introduction*

# *Equipment List and Setup – Vacuum chamber and pipework*

* 6-way cross vacuum chamber
  + Bottom flange: blank
  + Top flange: viewport
  + Front flange: scroll pump fitting
  + Rear flange: blank
  + Left flange: gas line entry and thermocouple
  + Right flange: pressure reader and air valve (to remove vacuum when needed)
* Scroll pump
* Copper end stage: copper block with a cut-out for curved metal gas pipe to sit, additional copper plate then placed over the top to sandwich the pipe within the block. Sits inside the chamber.
* K-type thermocouple attached to copper end stage, passes out of the vacuum chamber for analysis
* Pressure reader as above
* Air valve as above
* Foam bucket to hold LN2, with copper coil heat exchanger pipe sitting inside
* 6mm PTFE tubing with external foam insulation to carry cooled N2 gas. Runs from

copper coil heat exchanger output to the input of vacuum chamber, and from vacuum chamber output to the outside vent (open window)

* ¼ inch flexible tubing, makes up the remainder of the piping (prior to heat exchanger), handles room temperature N2
* 2x normally closed solenoid valves. ¼ inch tubing runs from N2 gas bottle to a junction of these two solenoids.
  + Solenoid 2 (S2) takes input N2 gas and bypasses the heat exchanger, taking room temp N2 gas to the other side of the heat exchanger and onwards into the vacuum chamber.
  + Solenoid 1 (S1) passes the input N2 gas to a mass flow controller (MFC). ¼ inch tubing then connects the MFC output to the copper coil heat exchanger input
* Alicat Mass Flow Controller (0-1000 SCCM)
* Unions, PTFE tape, nuts, bolts, washers, gaskets etc as required

# *Equipment List – Electronics*

* Arduino Uno R3
* AD8495 K-Type Thermocouple Amplifier breakout board.
* Arduino 4 relays shield (protects the Arduino, as the solenoids run on 24V)
* Adjustable power supply (to power solenoid valves with 24V)
* 12V power supply for the MFC
* Standard power supply for scroll pump
* “chocolate block” terminal connector strips of different sizes

# *Equipment Setup – Electronics*

* Arduino connected via USB-A to laptop (5V logic)
* Relay shield fitted to the Uno board using pre-integrated pins
* Terminal block soldered onto breakout board to enable thermocouple to be wired in
* 4x header pins soldered onto breakout board. Jumper wires used to connect this to the Arduino board
  + V+ connected to 5V Arduino pin
  + GND to Arduino GND
  + Output to chosen Arduino analog pin.
  + Additional GND connected to a common ground at the power pack to mitigate inductive interference (using chocolate block connector)
* Solenoids wired to relays 1 and 2 of the Arduino relay shield (V+ and GND) as well as into the adjustable power supply. Chocolate block connector used to achieve this.
* Manufacturer pinout used to wire connections for the MFC.
  + “Analog In” needed to set flow rate using input voltage. Wired back to PWM pin on Arduino
  + GND to complement this (wired to Arduino GND)
  + “Analog out” needed to read back the setpoint flow rate. Wired to analog input pin on Arduino
  + GND to complement this (wired to Arduino GND)
  + Additional GND connected to a common ground at the power pack to mitigate inductive interference (using chocolate block connector)

# *Equipment Setup – Software*

* Arduino IDE was used for initial tests and holds the majority of the hardware interface code. Required the installation of suitable libraries, detailed in code comments
* MATLAB is currently used to set the desired flow rate over serial connection to Arduino – however, the intention is to eventually run a semi-automated/regulated cycle in which the requirement for the user to set the flow rate is removed. It is likely that this connection will then just be used as a “go” button to start the experiment
* Data is sent from the Arduino to MATLAB over serial for data processing and visualisation. Currently this comprises the measured mass flow rate, temperature and the time at which each datapoint is recorded.
* The desired flow rate on the MFC is set using PWM on the Arduino (5V logic). The TimerOne library was used to enable higher frequency PWM than the Arduino default. This is important to avoid sampling errors, as the MFC samples flow rate at 1 kHz. The chosen PWM frequency is 10kHz.

# *Important Notes from Initial Testing – Thermocouple Breakout Board*

* The thermocouple breakout board has an accuracy of +/- 2°C within a specified temperature range according to manufacturer specifications. It also uses a manufacturer-specified equation to calculate temperature from the voltage differential:
  + However, this is affected by the actual output of the Arduino 5V pins, which may not be exactly 5V. This should be measured by a multimeter/voltmeter and the reference voltage in the formula adjusted accordingly. Due to the large amplification factor, even a small difference in reference voltage can have a large impact on the accuracy of the temperature reading.
  + Calibration testing with an additional K-type thermocouple and digital reader, using ice (0°C) and boiling water (100°C) can be used to trial offset adjustments
  + The length of the cable used to connect the Arduino to the laptop may also affect the reference voltage due to resistive losses in the cable wire. In this test setup a cable extender was used due to the spacing and placement of apparatus – this resulted in further adjustments being required
  + Adjustments needed are typically less than 1V.
    - Initially with the shorter Arduino connection cable, a voltage of 5.06V was read from the Arduino pins. This was adjusted by +0.6V to 5.12V for accurate readings
    - With the longer extension, a voltage of 5.02V was read from the Arduino pins. This required adjustment by +0.4V to 5.06V for accurate readings
    - Further details can be found in code comments
* Again due to the necessary large amplification required for the thermocouple voltage, the breakout board is especially vulnerable to inductive EMI. Even small voltage oscillations can be magnified by the amplifier and interpreted as a temperature reading. This caused significant issues in initial validation and testing.
  + Preliminary tests of the breakout board used a spare thermocouple (external i.e. not located inside the vacuum chamber) and the Arduino Uno board in isolation. Following calibration, these proved successful with the expected thermocouple readout
  + However, when testing the internal vacuum chamber thermocouple, significant interference was recorded in an oscillating cycle from around 5-40°C rather than reading around the ambient of 23°C.
  + After systematically turning on each electrical appliance in isolation and observing the effect on the thermocouple readings, the pump cable (running across the workbench next to the electronics) was found to be the primary cause, despite the pump only being on standby.
  + The MFC was the other source of EMI, but this was a significantly smaller effect due to the lower voltage supply (12V)
* To solve the problem of EMI, a common ground system was established where all relevant electronics were grounded to the ground of the adjustable power supply. This included the spare GND on the breakout board, the Arduino GND and a spare GND on the MFC pins. In addition, the pump power cable was rerouted to sit on the floor below the workbench
* After grounding, the oscillatory EMI was no longer present in the thermocouple readings. However, a small drop in accuracy was noted, with measurements sitting in a 5-6°C range rather than the specified 4°C from manufacturer. Reasons for this are unclear, but for this application this is not critical. Anomalous readings (currently defined as > 3 standard deviations from the mean in either direction) are easily removable in post-experiment data processing and analysis
* It should be noted that any semi-automated/regulated cycle which relies on temperature thresholds to activate certain processes (e.g. turn off solenoid 1 when minimum temperature is reached) is vulnerable to these accuracy deviations. Appropriate measures should be in place to prevent these from affecting the operation of the cycle (e.g. only turn off S1 when 5 consecutive readings below a threshold temperature are recorded)
* The pressure monitor has been known to also provide EMI, although not consistently. To avoid this impacting readings, it should be turned off during experiments.

# *Important Notes from Initial Testing – Mass Flow Controller*

* When setting the flow rate on the MFC, it was noted that low or high flow rates (towards the edge of the range) were often significantly different on the MFC reader than what was set in the code (e.g. a set flow rate of 50 SCCM was interpreted as 39 SCCM). This is likely due to the extremely short “on” (or “off” at the higher end) times associated with these flow rates.
  + This could be approximately corrected by adding a “fudge factor” of around 5 SCCM to the input flow rate. This is currently established in the Arduino code
  + It was also noted that the setpoint stabilised significantly when N2 was actually flowing through the device (compared to initial bench testing with no flow)
* Initial tests with room-temperature N2 at a regulated input pressure of 1.5 bar revealed a significant difference in flow rates between the bypass line and the flow-controlled line. In theory, at maximum set flow rate (1000 SCCM) the two should be similar, so this warranted further investigation
  + The bypass line experienced good flow and pressure throughout, with an audible sound of gas flow at the exhaust
  + However, the flow-controlled line had a significantly lower flow rate even when set to the MFC maximum of 1000 SCCM. Differences could be detected by setting the MFC to 50%, proving that the MFC itself was working, but the flows were so low that these were difficult to ascertain by sound/feel.
  + Possible reasons for this were investigated.
    - It was initially suspected that the copper coil heat exchanger could be throttling the flow due to a narrowed bore diameter at one end (present from manufacturer). This theory was tested by disconnecting the coil from the system and instead connecting a straight section of pipe of constant bore diameter. However, no improvement was noted so this was ruled out
    - A second theory was that the input pressure was too low for the MFC. However when checking on the datasheet, the minimum input pressure specified was 11.5 psia (equivalent to approx. 0.79 bar) so the test input pressure of 1.5 bar was clearly sufficient. In addition the MFC only experiences a maximum pressure drop of 1.5 psid (approx. 0.1 bar) so pressure along the pipe would have been little affected by its presence
    - The current theory is that the MFC was incorrectly chosen based on its flow rate range. When selecting the MFC, a simple online calculator was used to derive expected approximate flow rate from input pressure alone, assuming an operating pressure of 2 bar. This results in around 0.586 litres/min (or 586 CCM) – hence the decision to select the 0-1000 SCCM flow rate controller.
    - However, this was based on Poiseuille’s Law, which has two key assumptions: that the flow is laminar, and viscous. This means it is most applicable to liquids like water at low Reynold’s number. For gaseous N2, very slow velocities would be required for laminar flow, so it is almost definitely not laminar at the chosen input pressure. Gaseous N2 also has a very low viscosity which further invalidates the use of Poiseuille’s Law.
    - This means that the mass flow estimate under-predicted the reality of the gas flow, as the low viscosity results in higher flow speeds and consequently higher flow rates. So it is actually the MFC which is throttling the flow rate, even at its maximum of 1 SLPM. To solve this, a mass flow controller with a larger range should be chosen. However, this was not possible during the summer project duration due to long lead times on delivery. Instead, work progressed with the original MFC with the understanding that cooling times may take longer than expected due to lower flow rates

# *Important Notes from Initial Testing – LN2 Testing Campaign*

* A testing campaign was carried out to investigate the cooling curves produced by differing mass flow rates. Due to the MFC throttling discussed above, the first test was carried out at the maximum flow rate of 1000 SCCM (or 1 SLPM) with a test duration of 2 hours.