Science Module User Manuel 2022

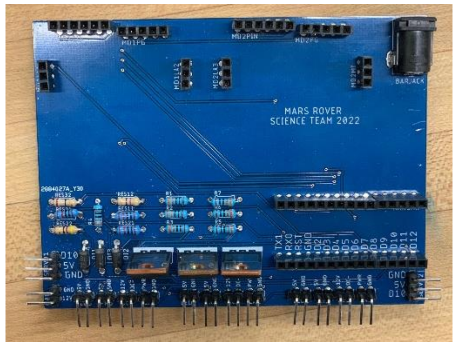
PICTURE OF FINISHED SCIENCE MODULE HERE!

Contents

1. Part Overview 2
   a) PCB 2
   b) Arduino 2
   c) Linear Actuators 3
   d) DC Motors 3
   e) Augers 4
   f) Vial Housing 4
   g) Biuret Vials 5
   h) Photoresistor Sensor 5
   i) FAD Microscope 6
   j) FAD Housing 6
   k) Microscope 6
   l) Microscope Housing 7
   m) Cameras 7
2. Setting up the Module to the Rover 8
   a) Wiring Components to the PCB 8
   b) Attaching the Module to the Rover Elevator 9
   c) Wiring the Module to the Rover 10
   d) Turning on the Rover 10
3. Software Overview 10
   a. Launch Code 10
   b. Arduino Code 10
   c. ROS Code 10
   d. GUI Code 11
   e. Debugging Tips 11
4. Biuret 11
5. FAD 12

1. Part Overview

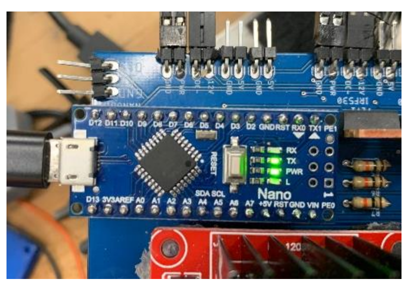
a) PCB



The PCB houses the Arduino and h-bridges and distributes 12v power and 5v logic to the rest of the science module. Specifically, it distributes power to the linear actuators, DC motors, servos, and photoresistor sensors. The following table provides the parts necessary to assemble the PCB.

Part Name	Quantity	Data Sheet Online
Barrel Jack	1	https://www.cuidevices.com/product/resource/pj-102a.pdf
H-Bride	2	https://components101.com/sites/default/files/component_datasheet/L298N-Motor-Driver-Datasheet.pdf
Arduino Nano	1	Google It
MOSFET	3	https://www.vishay.com/docs/91021/irf540.pdf
Diode	3	https://www.vishay.com/docs/88503/1n4001.pdf
Resistors	13	3.3kΩ, 300Ω, 10kΩ, 300Ω, 300Ω 3.3kΩ, 10kΩ, 300Ω, 300Ω 3.3kΩ, 10kΩ, 300Ω, 300Ω

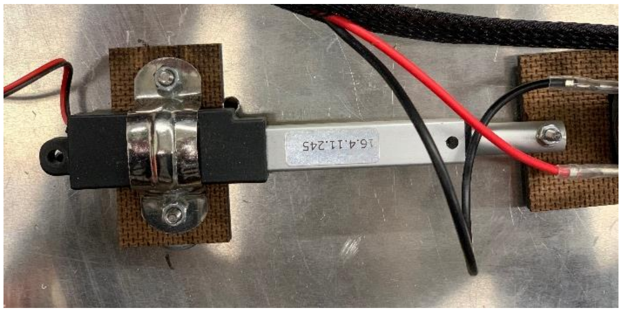
b) Arduino



The type of Arduino is Nano. It is attached to the module via the PCB. Care must be taken to ensure that the Nano on the science module does not confuse the rover computer by the rover computer confusing the science module Nano with the rover's Nano. The following details on UDEV rules can be found in I-001_Science_Information_For_Use:

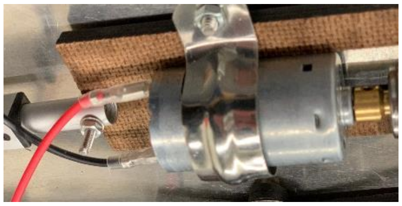
"If the devices don't seem to be connected properly (go to /dev/rover and type 'ls'), then try to reload the udev rules. Do this by going to /scripts/udev/ and typing 'bash reload.sh'. See if this fixes anything. If this does not fix anything, then further debugging may be required. When devices are plugged into Linux, they show up in the /dev/ directory (you can see this by typing in the commands “cd /dev” and then “ls”). The devices that we care about are in /dev/input/ (this is the xbox controllers which show up as jsX devices). Also, other USB devices show up as /dev/ttyACMX devices, or /dev/ttyUSBX devices. In order to read/write to them, you need to have the filepath to these devices. However, upon bootup, or plugin, the devices plugged in are mapped to a filepath upon various unpredictable factors. Thus, if an arduino shows up as ttyUSB2 one time, you can’t depend on the arduino being attached to the same ttyUSB2 the next time the Xavier boots up. So, what we use is instead we use symlinks and udev rules. Each device that is plugged into a computer has a list of associated attributes. (you can see these attributes by typing in the command ‘udevadm info –a –n <device filename>) The udev rules are located in /etc/udev/rules.d. All of these rules are run on boot time. Linux will look at the attribute rules in these rules files and try to match them with any devices that are connected to the computer. Compare the attributes list of a device with the corresponding rule.”

c) Linear Actuators



The Linear Actuators are powered by the 12v distributed from the PCB. They can be thought of as linear DC motors, i.e. applying a voltage on one lead causes extension and applying a voltage on the opposite lead causes contraction. There are 3 linear actuators on the module.

d) DC Motors



The DC motors turn the augers and are powered by the 12v distributed from the PCB. There are 3 DC motors on the module.

e) Augers

The augers are used to deliver soil samples to the Biuret vials. They are turned by the DC motors and must be turned in the positive direction indicated by the right-hand-rule with the thumb extended away from the rover.

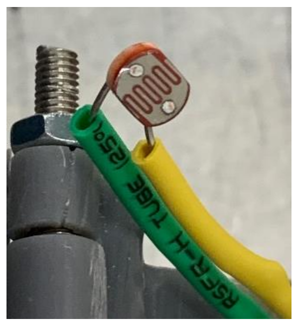
f) Vial Housing

The vials of Biuret must be housed in a container that blocks out light and allows delivery of soil from the augers. CAD models of the current vial housing can be found on the BOX in the “Team 49 2021-2022”->”Science_Subteam”->”ScienceModuleCAD” directory.

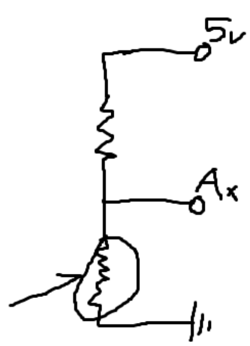
g) Biuret Vials

The Biuret Vials hold a mixture of Biuret and water. The containers must be clear to allow light from the photoresistor sensor to pass through. See 4. Biuret for more information on the chemical process of Biuret.

h) Photoresistor Sensor



There are three photoresistor sensors on the module. The photoresistor sensor contains two components: the photoresistor and the green LED. The two components are placed on opposite sides of the vial housing so that light from the LED passes through the Biuret vial and onto the photoresistor. The LED is powered by the 5v distributed by the PCB. The photoresistor is one part of a voltage divider that changes the voltage applied to an analogue pin on the Arduino as shown in the following wire diagram:

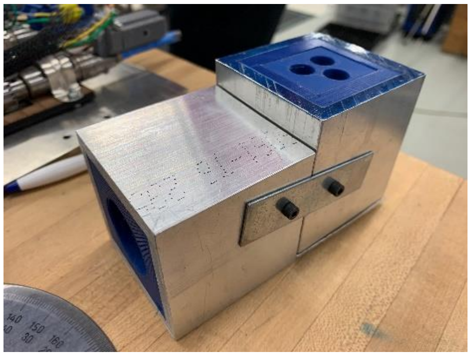


The normal resistor must have large resistance ($\approx 10k\Omega$) to ensure dramatic change of voltage applied to the Arduino's analogue pin.

i) FAD Microscope

The FAD microscope is used to observe extant life on soil or rock samples. It is powered by USB from the rover computer. See FAD Housing for more details on FAD life detection and 5.FAD on information about the chemical process.

j) FAD Housing

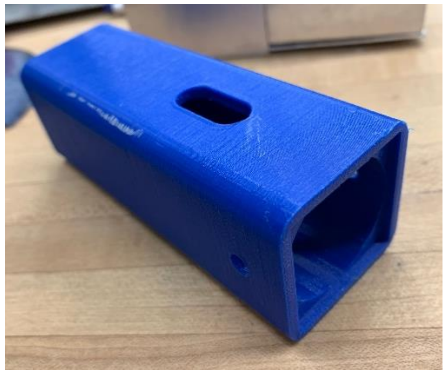


The FAD housing does more than hold the FAD microscope. It also contains 3 blue LEDs powered by the 12v distributed from the PCB and a light filter donated from Chroma with the following details https://cn.chroma.com/node/26719. Essentially, the blue LEDs emit light which interacts with extant life which emits a phosphorous green which is made manifest through the light filter. See 5.FAD for details on the chemical process.

k) Microscope

The Microscope is used to detect extinct life on rock samples (fossils). It is powered by USB from the rover computer. Note that it is very difficult for untrained eyes to identify fossils using this instrument and it is thus recommended that a knowledgeable geologist should be enlisted.

l) Microscope Housing



The microscope is housed in a container that interfaces with the science module. Nothing more.

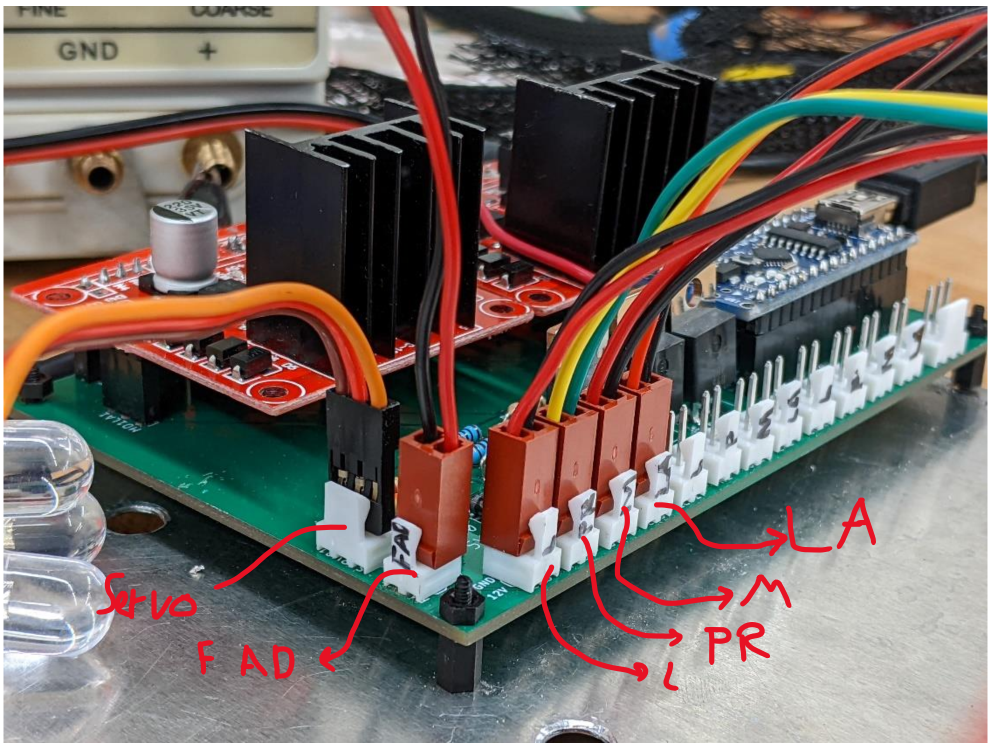
m) Cameras

Various cameras are used to provide visual feedback to the rover driver. They can also be used to help identify extinct life on rock samples (fossils). They are powered by USB from the rover computer. (MORE ON CAMERA LOCATIONS AND TYPE HERE).

2. Setting up the Module to the Rover

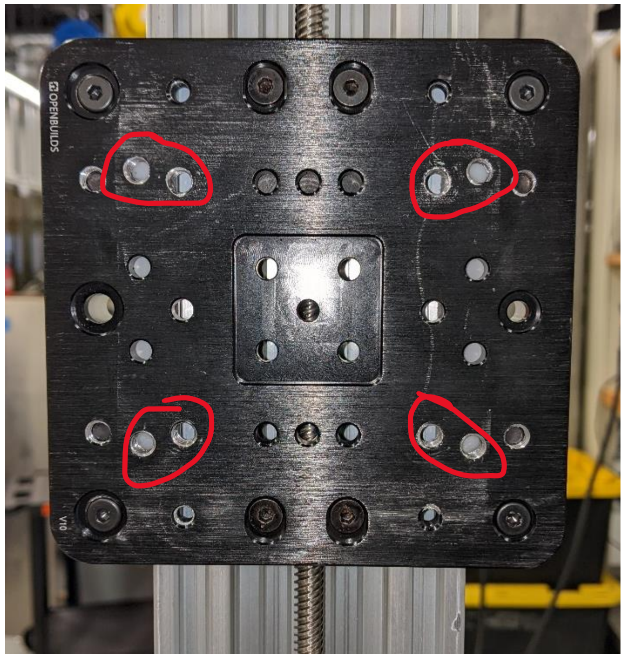
a) Wiring Components to the PCB

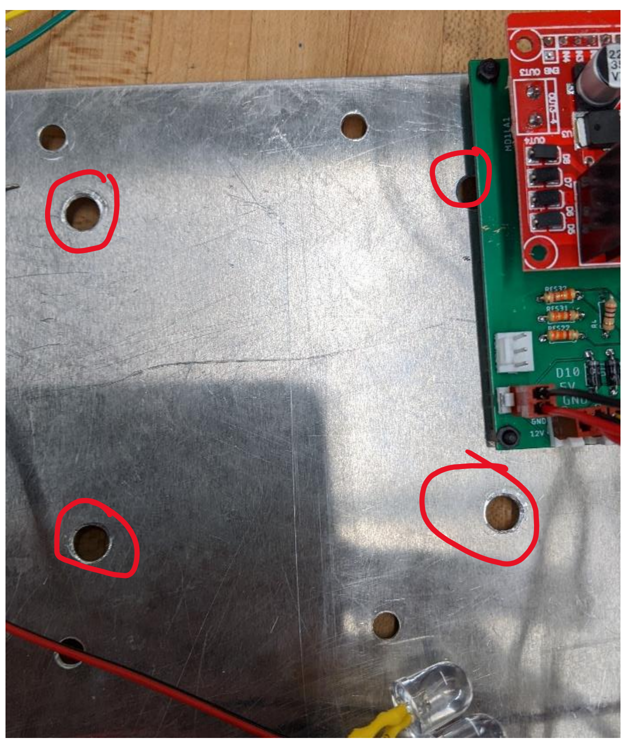
There are 17 components that are attached to the PCB, specifically, the 12v power plug into the barrel jack; the USB connection to the Arduino, two servo motors, three DC motors, three photoresistor sensors, three linear actuators, and 12v power output for the blue LED's on the FAD housing. On the long side of the PCB are three sets of wiring. From left to right the wiring pattern for each set is LED, Photoresistor, Motor, and Linear Actuator. As seen in the following picture there is also an FAD power plug and the two three pin connections are for servos.



b) Attaching the Module to the Rover Elevator

The backplate of the science module has four mounting holes which interface with the elevator of the module.





c) Wiring the Module to the Rover

There are NUMSTUFFHERE! Components that are attached to the rover, specifically, the USB of the FAD microscope, the USB of the Arduino, the USB of the microscope, the USB of the cameras, and the 12v power supply.

PICTURE OF SETUP HERE!

d) Turning on the Rover

3. Software Overview

a. Launch Code

A helpful diagram of the launch files can be found on the git wiki (https://github.com/BYUMarsRover/BYU-Mars-Rover/wiki/Code-Diagrams).

The rover is launched from the base station. Navigate to the BYU-Mars-Rover/scripts directory in the terminal. Run the launch.sh file with the following:

bash launch.sh -t science

The syntax is as follows:

bash launch.sh <rover ip address: defaults to 192.168.1.20, local runs locally> -t <task: autonomy, retrieval, servicing, science>

The launch script starts a tmux server with the left pane running locally and the right pane running on the rover (which is determined by the given ip address). After doing this, a launch file corresponding the given task is launched on the base station as well as the rover. These files are found in BYU-Mars-Rover/rover_ws/src/start. Each task launch file also runs a common launch file for the base station and one for the rover. Each of these launch files should be launching the ROS nodes needed for each branch of the competition.

After launch, the bottom of the tmux panel shows which ROS nodes are running. You can input the corresponding character to bring up a menu to start, stop, or mute the ROS node. This can be helpful for frequent minor changes because changes to code are applied when stopping and restarting a ROS node. This allows for minor changes to be implemented without a complete restart of the base station and rover.

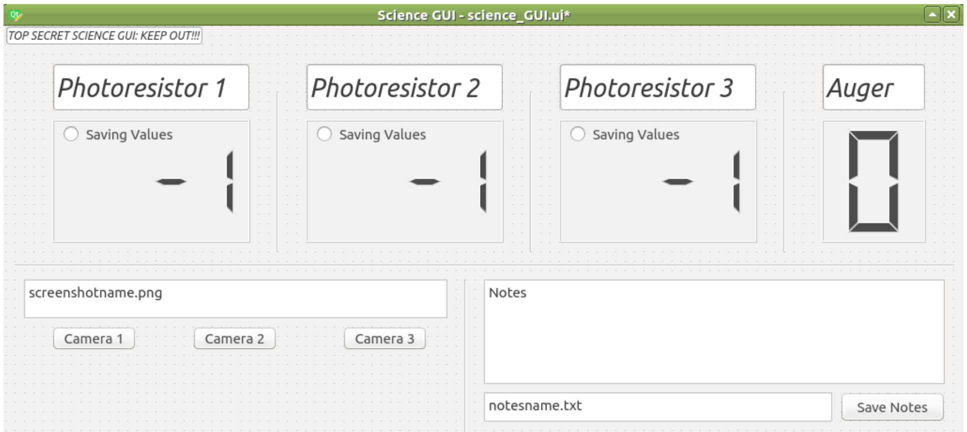
b. Arduino Code

The Arduino code is located in the team's git repository (BYU-Mars-Rover/rover_ws/src/science/src/science_arduino/science_arduino.ino). It takes in values (over serial) for the linear actuator directions and auger directions. It outputs (again over serial) the photoresistor values. The Arduino code also turns on the LED's and reads in the photoresistor values.

c. ROS Code

A helpful diagram is found on the wiki (https://github.com/BYUMarsRover/BYU-Mars-Rover/wiki/Code-Diagrams). The ROS code takes in values from the XBOX controller; decides on directions for the linear actuators, augur motors, and elevator; sends those values to the corresponding Arduinos; and reads photoresistor values from the science Arduino. It also sends the photoresistor and current augur index values to the science GUI.

d. GUI Code



The science GUI displays current photoresistor values and current augur index. The GUI also controls which cameras take screenshots, recording data from the photoresistors, and saving notes.

e. Debugging Tips

There are a couple of common bugs and ways to find them. If ROS message imports are not found, then running a catkin_make will likely fix the problem (scripts are located in BYU-Mars-Rover/rover_ws)

A good way to debug ROS topics is by opening another terminal and running **rostopic list** (for a list of topics) and **rostopic echo <topic>** (to see what messages the topic is receiving). If a topic is receiving messages, but a corresponding subscriber is not, then make sure that the subscriber's file is in a launched node, and the class has been instantiated. If that doesn't work, then there might be an issue with the ROS environment when a topic is not being shared from the base station to the rover and vice versa.

Print statements are your friend. They will appear in the terminal. If a node is being too "noisy" then you can mute it by inputting the corresponding character and selecting the mute option in the terminal. When finished debugging, just make sure to remove or comment out your print statement.

4. Biuret

Biuret Reagent is used to detect peptide bonds in a solution. The solution starts from a light blue, and proportionally transitions to a purple with the concentration of peptide bonds. The wavelength that gets absorbed in the reacted purple reagent is approximately 540nm, a green. The BYU Mars Rover Biuret Test puts a soil sample into a [vial](#) of reagent and then shines a green LED directly on a photoresistor with the biuret in between. If there is life in the sample, the color will change to purple and absorb the LED light which decreases the reading on the [photoresistor sensor](#). The following article explains this process very well: https://www.mycrobe.org/blog/2018/7/26/biuret-assay. The following article is the original rediscovery of the reaction by Dr. Piotrowski in 1857: http://www.biodiversitylibrary.org/page/6896337.

5. FAD

Flavin adenine dinucleotide (FAD) is a coenzyme that has an electron excitation state that emits light at about 370nm, a green. The idea is to excite the sample with strong blue light and then, filtering out the blue light, observe if any green is detected. The following article gives a very detailed description of FAD and its application: https://zaguan.unizar.es/record/70639/files/texto_completo.pdf?version=1.