We observed very soon into the project that the IR sensor does not give consistent readings above the floor of the Design Studio, where the craft is meant to operate, thus rendering this sensor useless for our application. We checked the sensor against other surfaces, to make sure it functions correctly, and we found that it indeed works accurately and consistently over other surfaces. We would have preferred the IR sensor over the ultrasonic one, because, being an analogue sensor, the reading time of it would have been only limited by the Arduino processing speed, which from testing was found to be around 2-3 ms, while the ultrasonic sensor has a much longer reading time, as sounds travels a lot slower than light and because we have to wait to make sure the previous sound have doesn’t echo back after a new one was emitted as that would give readings that are very far out from reality. This means that we must slow down the reading frequency of the sensor to 10Hz, to ensure enough time has passed before sending another sound pulse.

From the beginning, we decided not to use an Arduino UNO as our autopilot, but to go with a smaller, lighter version of this, which is the Arduino Nano. The boards are very similar in terms of their functionality, both using the same microprocessor. This allows us to code the boards in the very same way, from the same software. This modification presented us with a few other redesigns.

Firstly, we were not able to use anymore the GPS shield to store telemetry data from our aircraft. To solve this, we decided to use Bluetooth to transmit data to our laptops. One Bluetooth module is situated on our craft, which is encoding data containing: current altitude, throttle, autopilot state and grabber state, into a single string of characters, that can be sent all at once, and transmits them to another Bluetooth module, which is hooked up to another Arduino that decodes the data, before sending the data to a laptop, where it is store in an Excel file. the functions that encode and decode the data can be seen in the figures 1 and 2 in the appendix. This way the data can be viewed in real time, and graphs that describe the behavior of the craft can be plotted with ease. Encoding and decoding the data into a single string of characters is the easiest way we have found to synchronize the transmission and the reception of the correct data points at any given time. Currently data is being transferred from the craft to the laptop at a frequency of 2Hz, which is enough for analysis, as the PID primary loop, which includes the PID loop that controls the autopilot, only runs at 10Hz. As a back-up to our Bluetooth solution, we are able to fit a SD card module onto the craft to store data onboard, which can be processed later.

Secondly, Arduino Nano’s are designed to be mounted on an either a copper board or on a breadboard. We chose a copper board as the circuit can be minimalized, both in terms of size and in terms of weight, to save space on the craft. The circuit includes all connections required with the receiver, flight controller, ultrasonic sensor and Bluetooth module. It also includes a relay, whose function is to switch between autopilot and manual flight. A relay was chosen as it provides a physical connection between the receiver’s throttle channel and the flight controller, meaning that if the Arduino fails, in any way, or if the power to the Arduino is lost, then control of the craft can still be maintained as the relay falls back into it’s normal state, which is designed to be manual mode. At the same time, if the Arduino losses signal from the receiver, the relay falls into the same, providing manual control of all channels. Although a transistor was considered to switch the relay, the datasheet of the part indicated that the coil only draws 30 mA at 5V, which is the Arduino voltage, while the I/O pins of the board can provide up to 35 mA, indicating that the relay can be powered directly from the Arduino, thus saving more space and simplifying the circuit more. In retrospective, a fly-back diode should have been added to cope with the surge in voltage that occurs when the relay is powered off. An LED is also present on the board that lights up to confirm a stable Bluetooth connection between the craft and the other Arduino. A schematic of the circuit can be seen in figure 3, together with a copper board layout.

Appendix

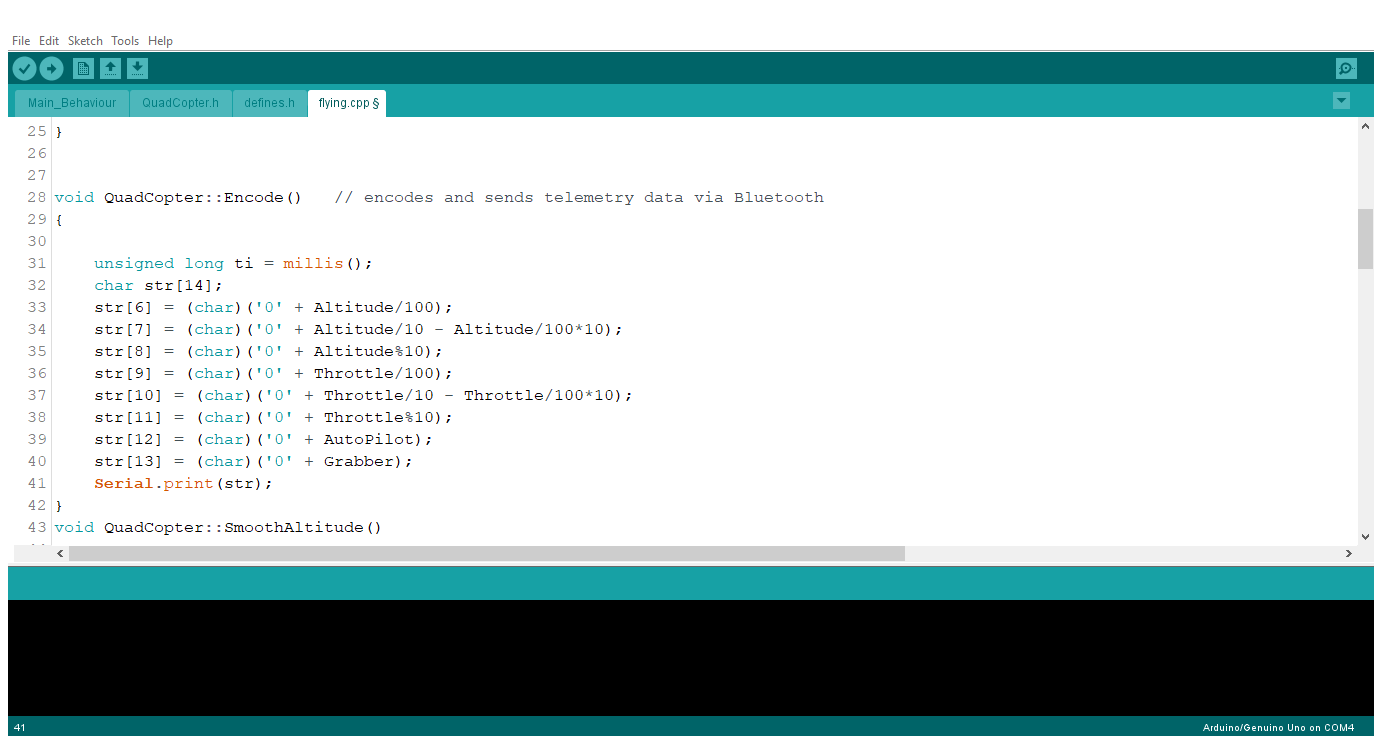


Figure Data encoding function

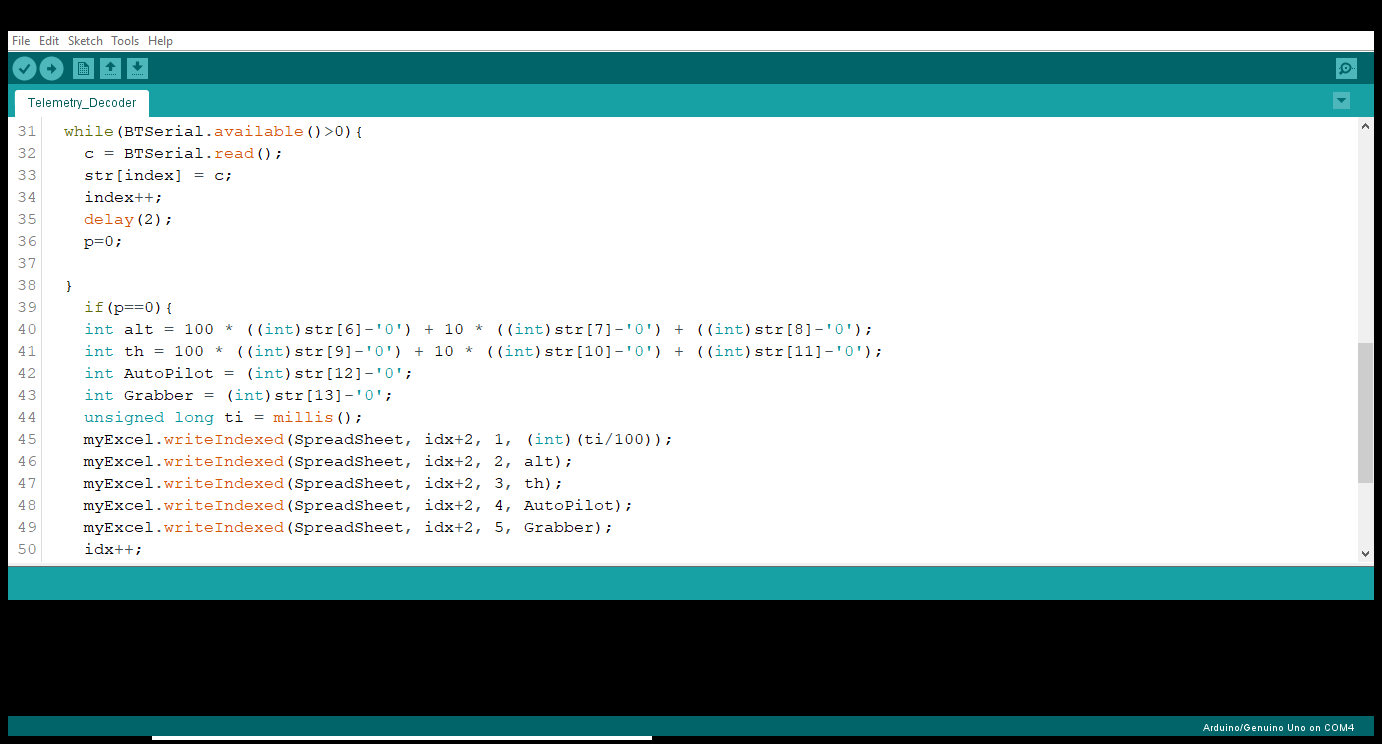


Figure Data decoding function

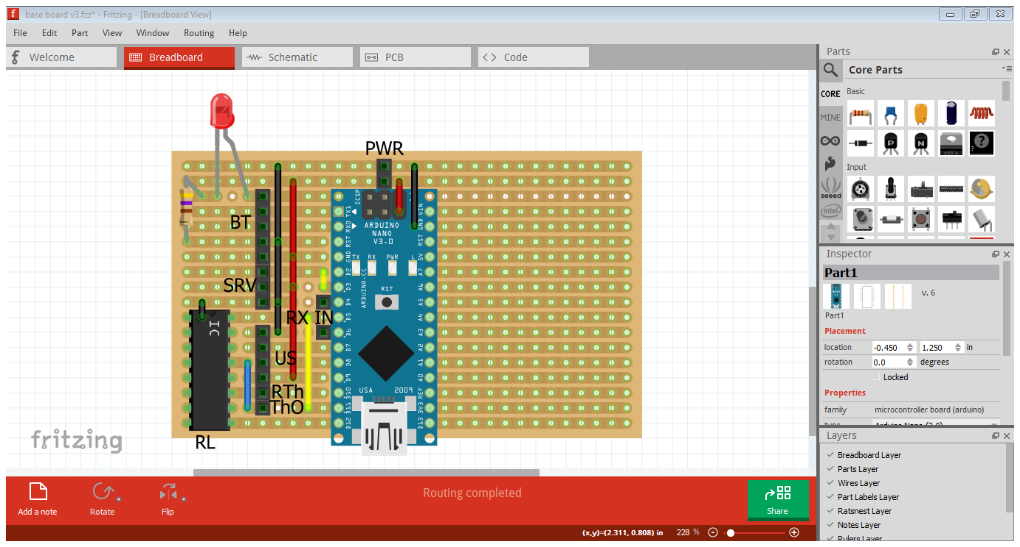
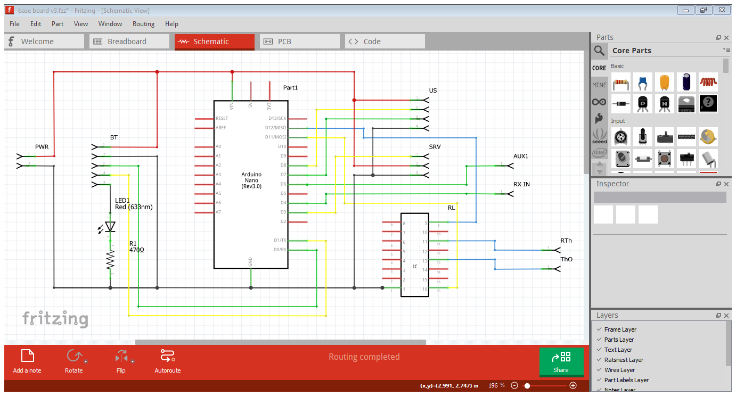


Figure Electronic schematic