Milestone 4
Detailed Design Checklist

Richard Boone, Kyle Douglas, Sayali Kakade,
Sang Min Oh, & Aditya Wadaskar

I. INTENDED SOFTWARE STRUCTURE

OVERALL STRUCTURE

The software structure of each part of the project is based on what we plan to demonstrate at the Design Expo in May 2019. The following list describes our plan for the final demonstration:

1. Both the parent and child drones are at the base station.
2. The child drone is flown to point A using radio control. The child remains stationary and hovers at point A while the parent drone is still at the base station.
3. The parent drone is activated – it is now able to communicate with the child drone over WIFI.
4. The child communicates its GPS coordinates (point A) to the parent.
5. The parent drone flies to N feet directly below the child drone at point A.
6. The child drone detects the parent drone, descends, and lands on the surface of the parent drone.
7. The parent drone replaces the child drone's stale battery.
8. Once the battery-switching is complete, the child drone unlatches from the surface of the parent drone and flies back to point A.
9. The parent drone is flown back to the base station using radio control, where we replace both its battery and a full battery for the child drone.
10. We repeat steps 3-9. The entire time, the child drone is hovering at point A.
11. If we want to land the child drone, we can take control of it and bring it back to the base station using radio control. If desired, we can restart the demonstration from step 1. Furthermore, there will be battery indicators for both the parent and child drones in case of emergencies.

Parent Drone

The parent drone has one on-board computer, a Raspberry Pi 3 B+, which will run Raspbian Lite (a headless Debian-based Linux distribution). The following table outlines the hardware-software interactions and the layout of the software with respect to each of the modules that will need to be programmed for interaction with the Raspberry Pi.

Hardware Connections to Software	**Purpose of Software**
USB1 → DJI N3 Flight Controller	UART connection: sends directional instructions to the parent drone to control its movement. We will explore the exact interfacing and API calls further.
USB2 → Wifi Adapter	USB connection: acts as a fail-safe for the DGPS module. It allows remote access to the Raspberry Pi for other communications between the parent and child drones, if necessary.
USB3 → ublox DGPS Module	USB connection: provides RTK positioning, with the parent drone acting as the moving baseline in relation to the child drone.
GPIO Pin 2 → Linear Actuator	Applies a linear force to move batteries in and out of the battery-switching contraption on the child drone.

The structure of the software on the Raspberry Pi 3 B+ will be as follows:

State 1: Deactivated – The parent drone is not searching for the child drone

```
while parent drone not activated do
|   ignore child drone;
end
transition to state 2;
```

State 2: Activated – The parent drone is actively searching for the child drone

```
while no communication with child drone do
|   establish communication over WIFI with child drone;
end
retrieve GPS coordinate of child drone;
fly to N feet below child drone and hover;
activate electromagnets in preparation for the child drone landing;
while child drone not latched to parent do
|   hover in place;
end
transition to state 3;
```

State 3: Battery Switching – The child drone is latched onto the parent drone

```
activate linear actuator;
insert new battery into child drone and push out stale battery;
signal child drone to power on and unlatch from parent drone;
while signal acknowledgment not received from child drone do
|   keep electromagnets activated;
end
deactivate electromagnets;
while child is latched onto parent do
|   hover in place;
end
transition back to state 1;
```

Child Drone

The child drone will also have an on-board computer, a Raspberry Pi Zero W, which will also run Raspbian Lite. The child drone will have a PixRacer flight controller, which will be running the px4 flight control framework. The following table outlines the hardware-software interactions and the layout of the software for the child drone's on-board computer.

Hardware Connections to Software	Purpose of Software
USB $\rightarrow$ Telemetry1 of PixRacer	Serial port connection: controls the movement of the child drone motors using directional commands.
GPIO 4, 6 $\rightarrow$ Step-Down Converter	Steps-down the battery voltage from 14.7 V to 5 V for powering other peripherals.
GPIO 8, 10 $\rightarrow$ OpenMV P4/P5 (TX/RX)	I^2C connection: communicates the location in which the AprilTag (and thus, the parent drone) is detected.
UART $\rightarrow$ ublox DGPS Module	Gets the current location of the parent drone with the child drone acting as the rover in the ublox RTK moving baseline model.

The structure of the software on the Raspberry Pi Zero W will be as follows:

State 1: Deactivated – The parent drone is not searching for the child drone

```
while parent drone not activated do
  | remain stationary;
end
transition to state 2;
```

State 2: Activated – The child drone is hovering and waiting for the parent drone

```
while AprilTag not detected do
  | hover in place;
end
while child drone not latched to parent drone do
  | read AprilTag information from OpenMV;
  | provide directions to PixRacer to descend and land on parent drone;
end
signal to the parent drone indicating successful latch;
transition to state 3;
```

State 3: Battery Switching – The child drone is latched onto the parent drone

```
while signal to unlatch not received from parent drone do
  | remain stationary on top of the parent drone;
end
supply power to motors;
send acknowledgment of signal to parent drone;
rapidly take off from the parent drone;
transition back to state 2;
```

II. Schematics

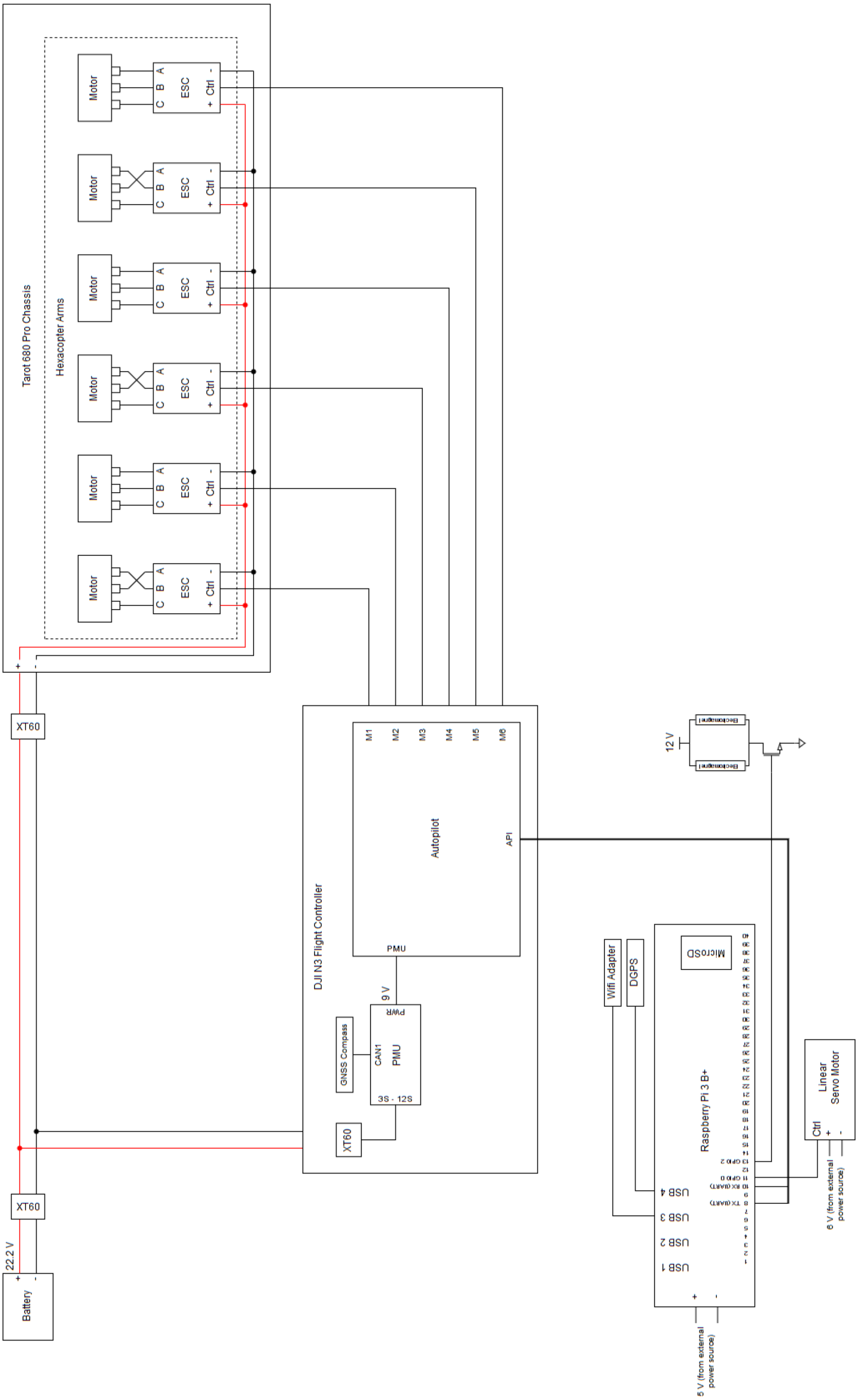

Figure 1: Parent Drone Schematic

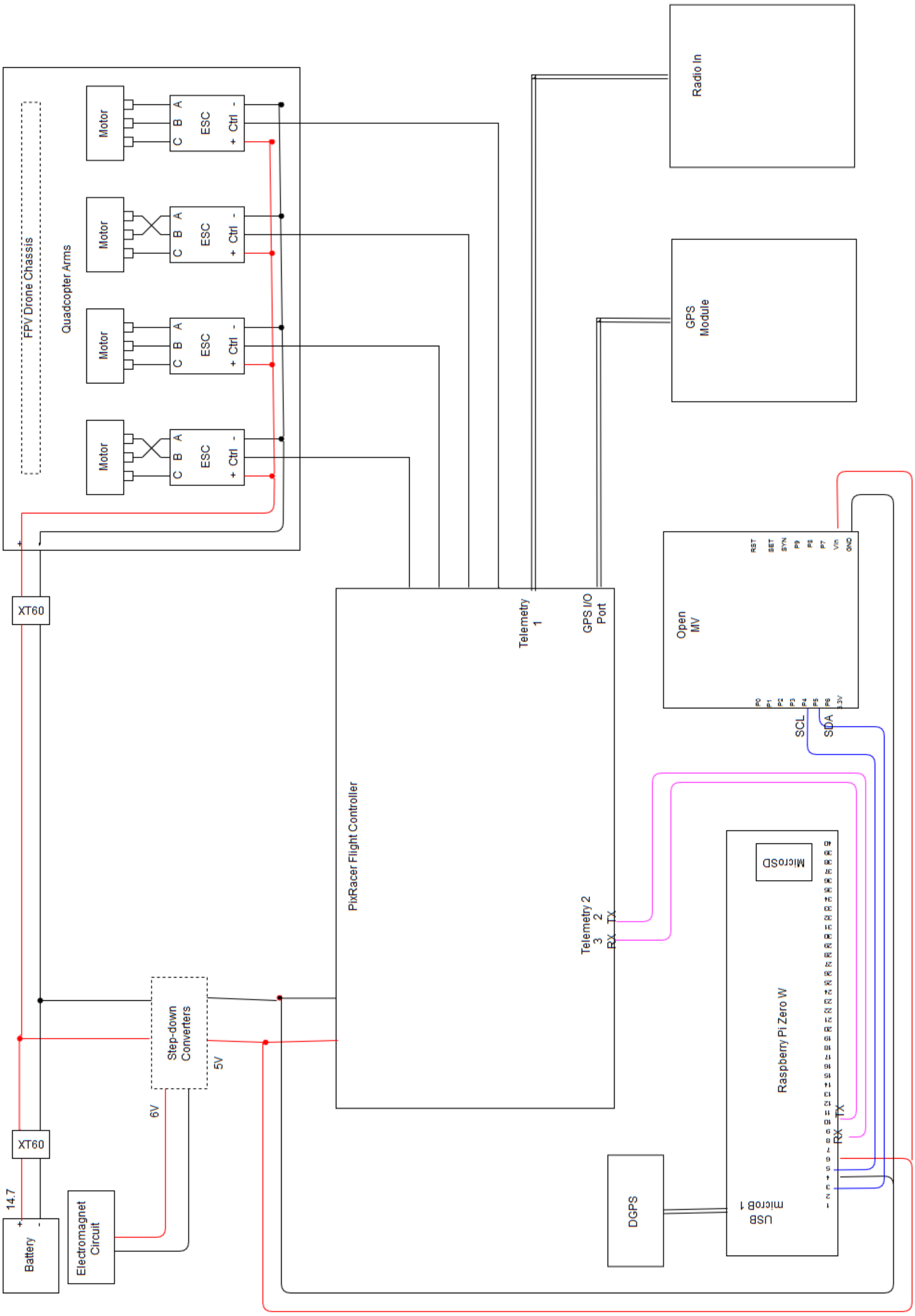

Figure 2: Child Drone Schematic