Milestone 4
Detailed Design Checklist

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

I. INTENDED SOFTWARE STRUCTURE

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 $\rightarrow$ 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 $\rightarrow$ 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 $\rightarrow$ 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 $\rightarrow$ 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 → Telemetry1 of PixRacer	Serial port connection: controls the movement of the child drone motors using directional commands.
GPIO 4, 6 → Step-Down Converter	Steps-down the battery voltage from 14.7 V to 5 V for powering other peripherals.
GPIO 8, 10 → OpenMV P4/P5 (TX/RX)	I^2C connection: communicates the location in which the AprilTag (and thus, the parent drone) is detected.
UART → 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

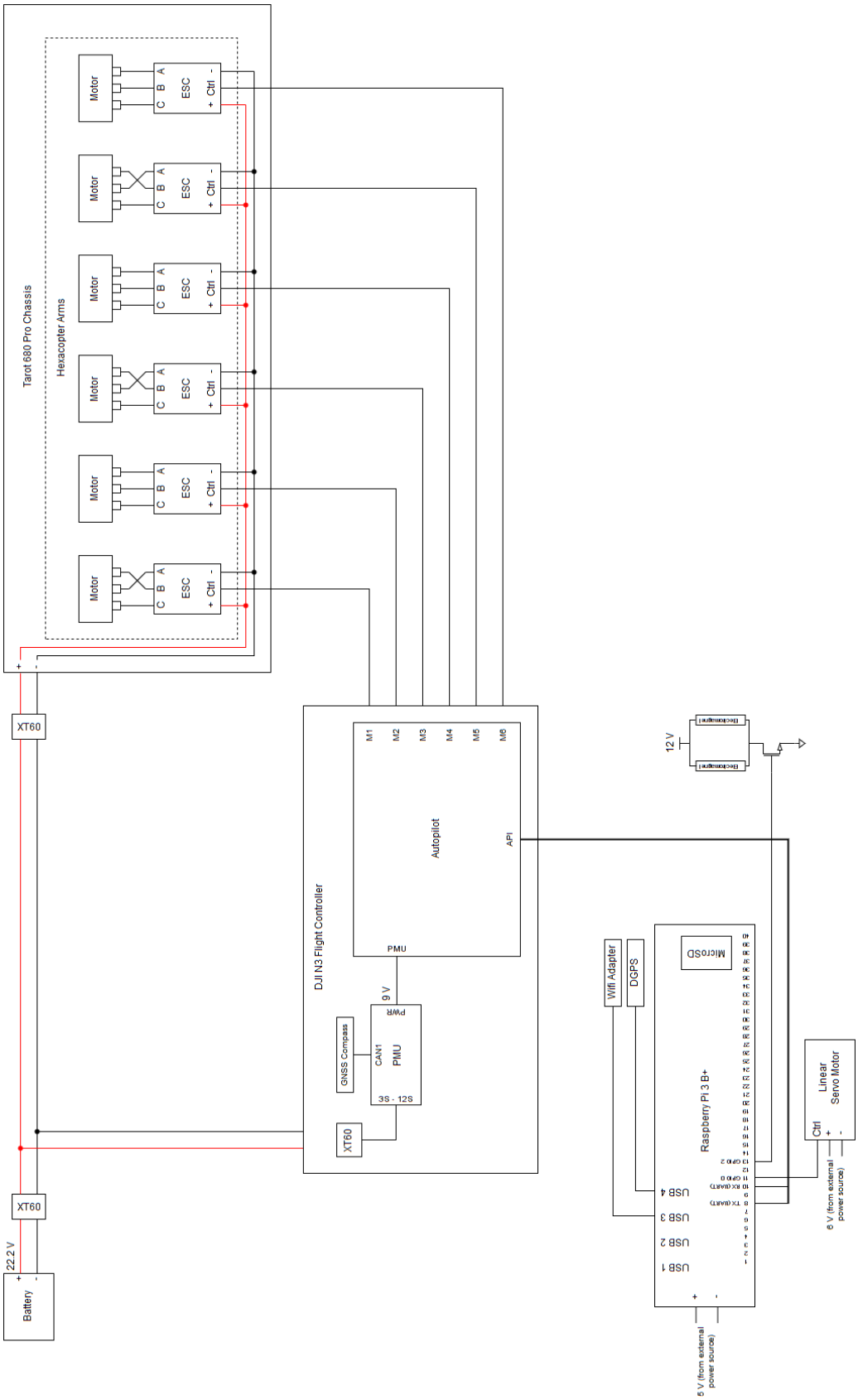

Figure 1: Parent Drone Schematic

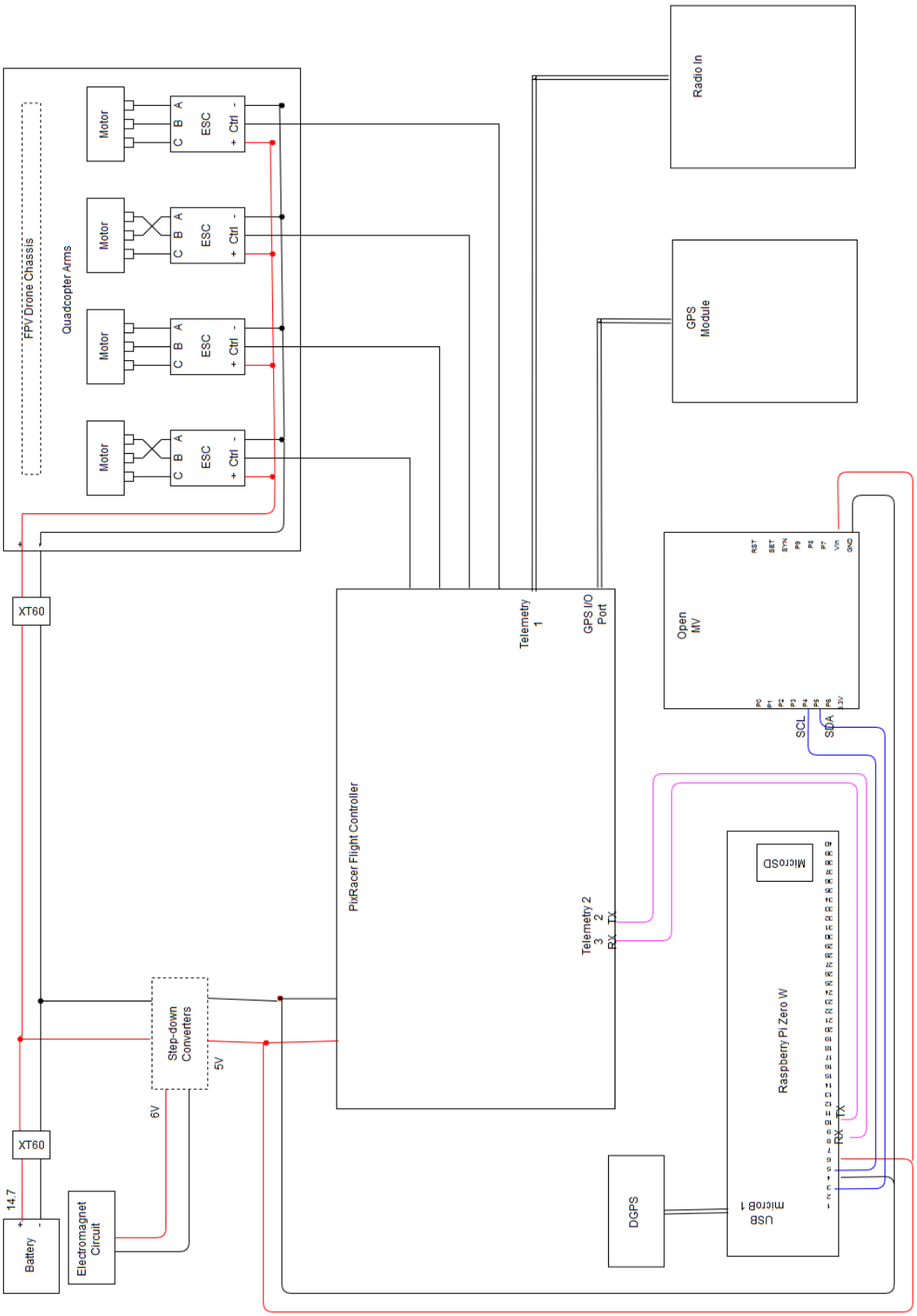

Figure 2: Child Drone Schematic