Robomagellan Autonomous Robot
6 Wheel Drive Linux

What is Robomagellan? Why do this?

Robomagellan is a robotics competition: Robothon, Robogames

- The robot must navigate a course (unstructured pedestrian terrain)
- The robot is given a set of locations 30 min. before the race
- The robot navigates to the locations in any order
- Only the final (finish) location is mandatory
- At each location the robot must find and make contact with a cone
- Contacting cones at intermediate locations boosts robot's score
- The robot can't damage the environment or hurt people (obviously)
- The robot has size and weight limits <= 4'x4'x4' and 50lbs
- No internal combustion engines, self contained, autonomous
- No differential GPS, no active external localization other than GPS

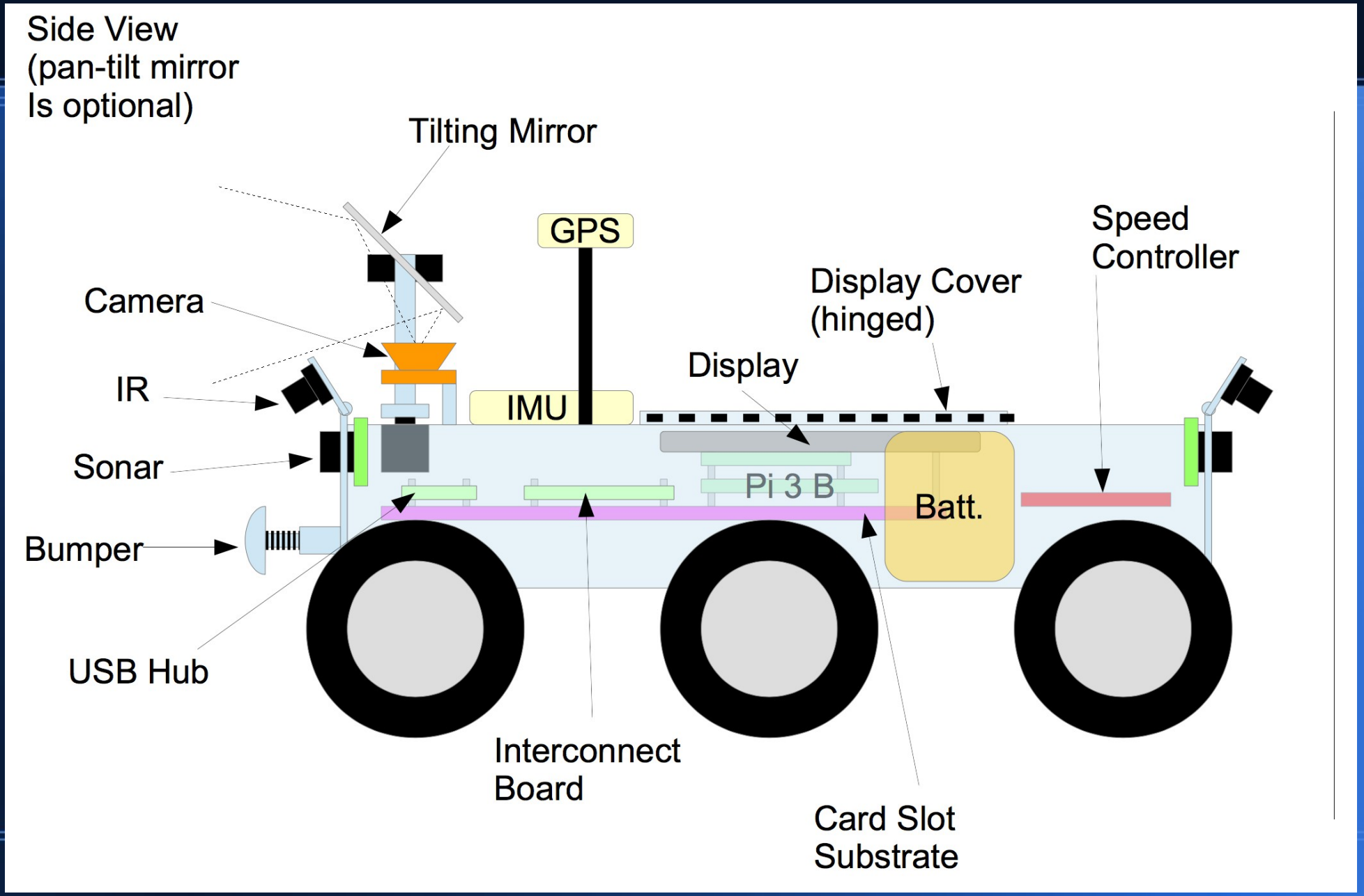
What do you get when you win? Bragging rights, a medal?

Objective

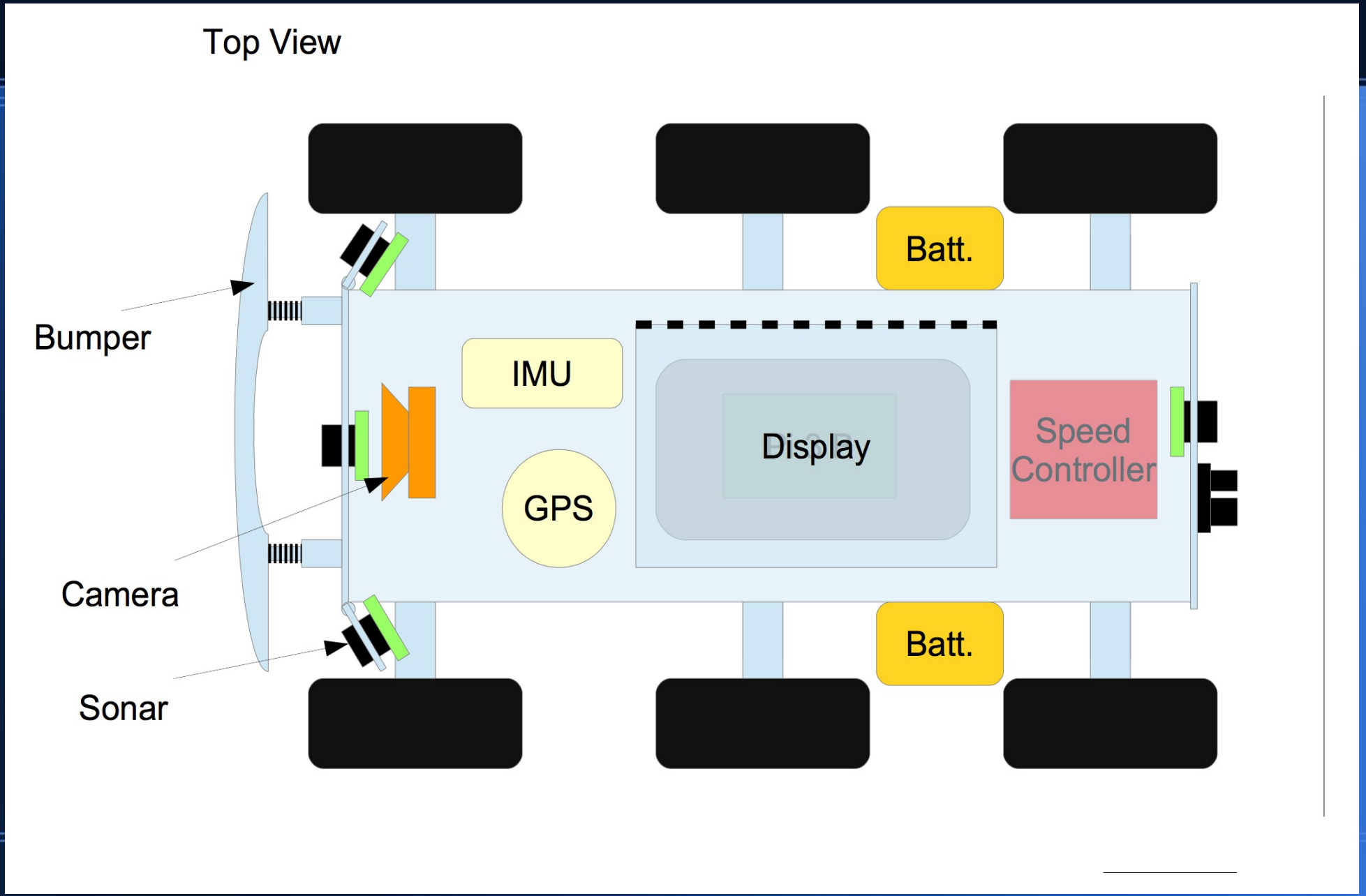
Design and build a robot capable of the following:

- Autonomy, (no human control except kill switch)
- Outdoor navigation in unstructured environment
- Collision avoidance, obstacle detection
- Vision, target object detection and tracking
- Contact sensing of target object
- Running 100% free and open source software (Linux!)

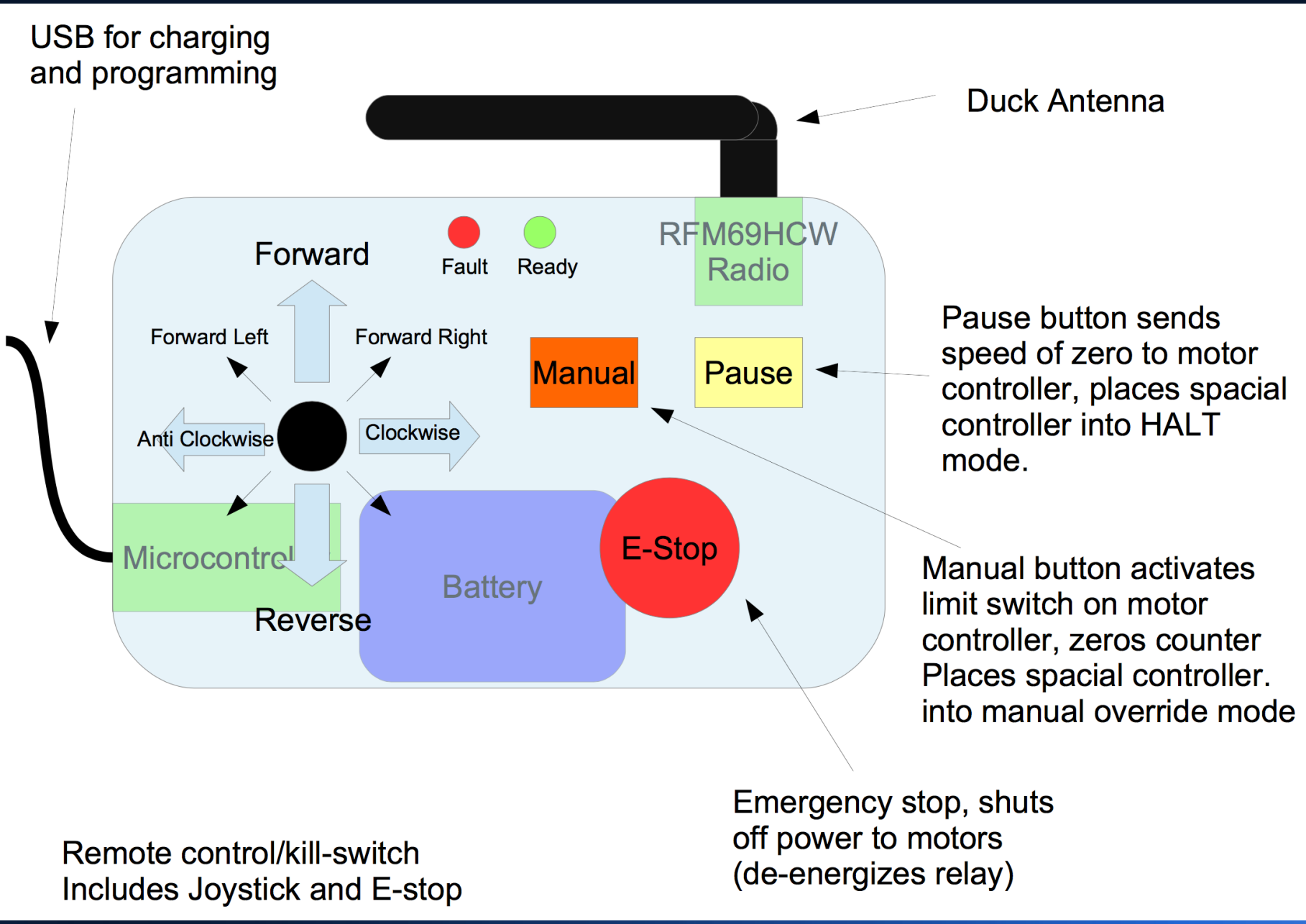
Platform Drawings



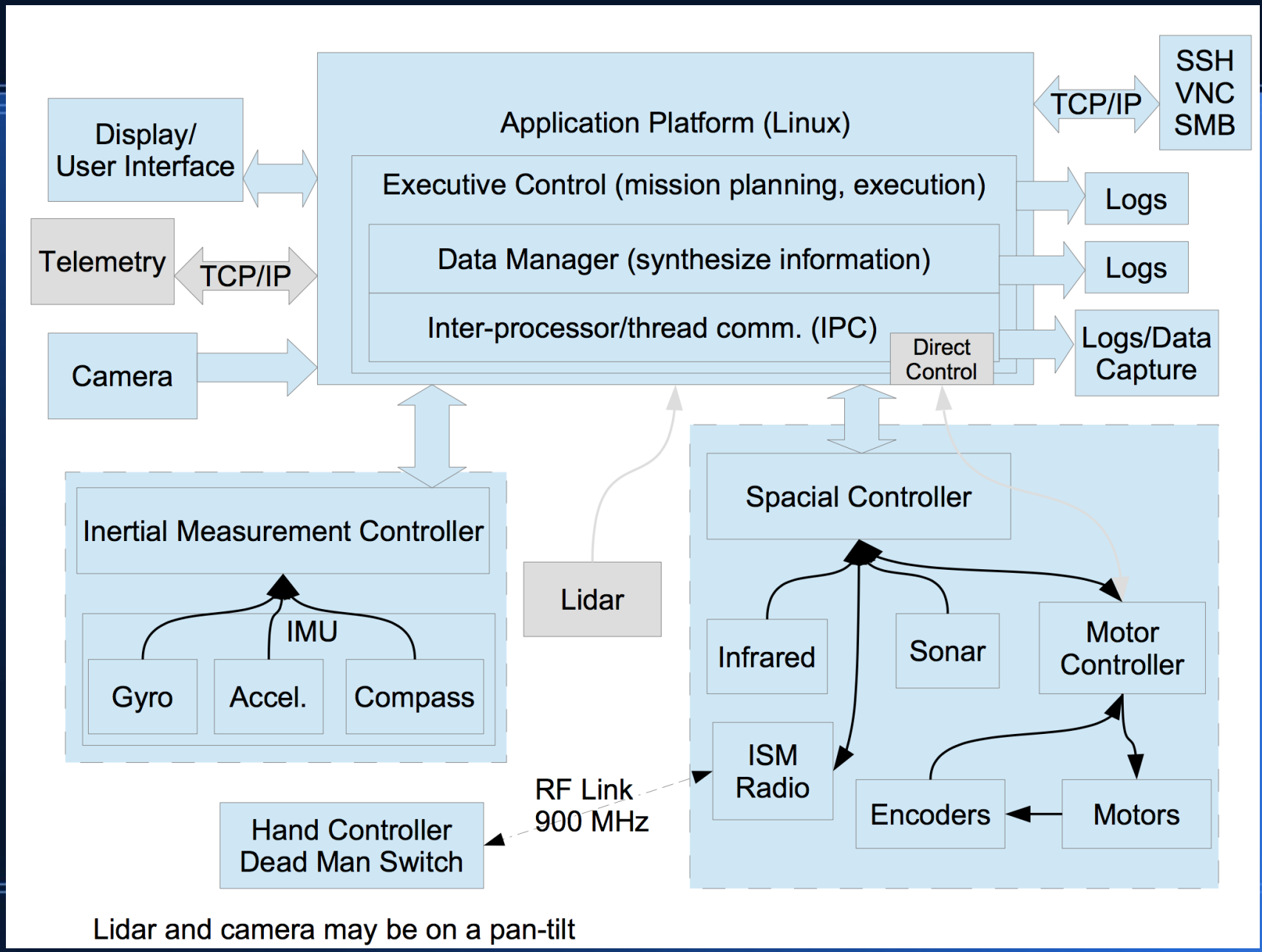
Platform Drawings Continued



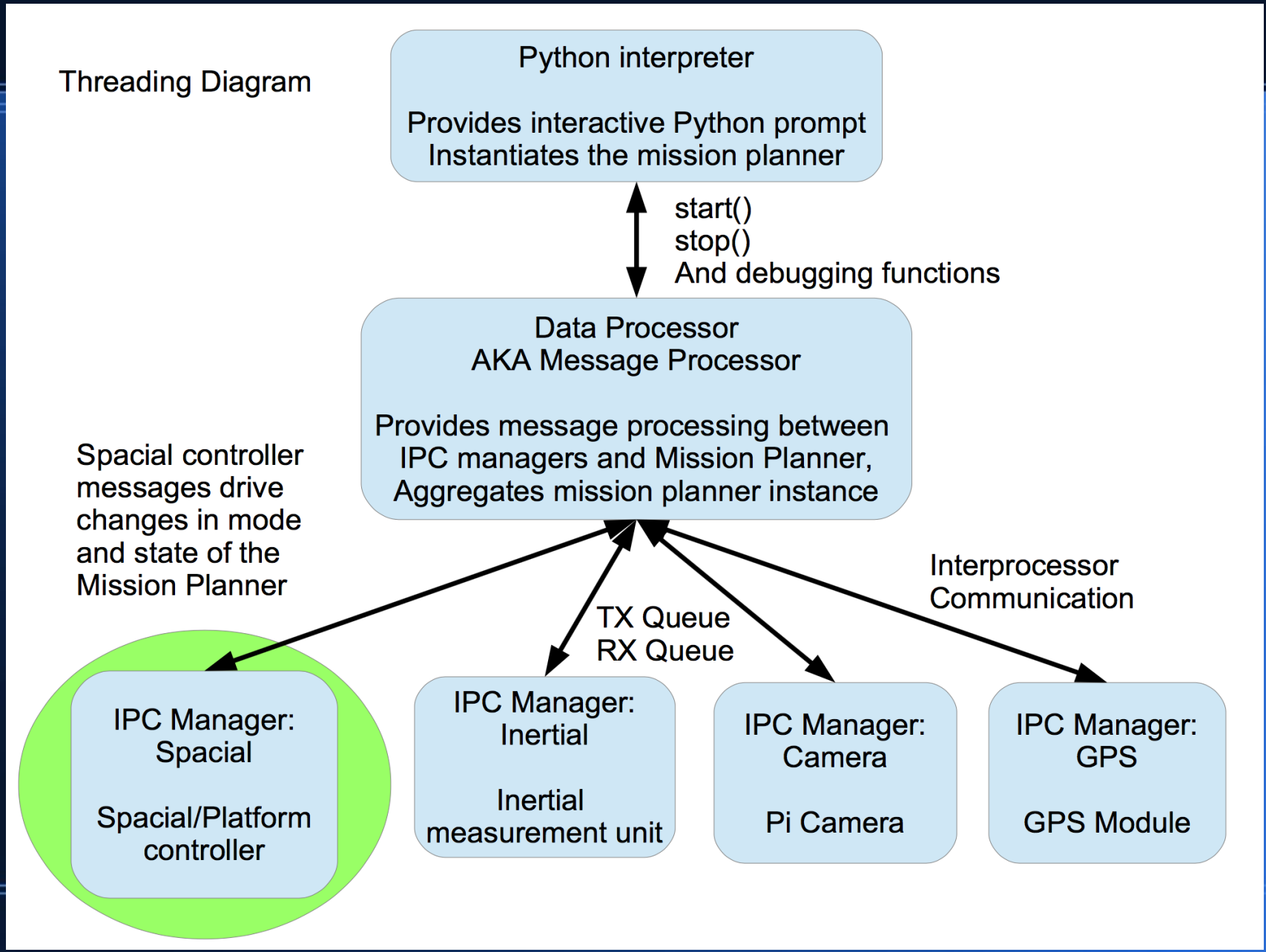
Platform Drawings Continued...



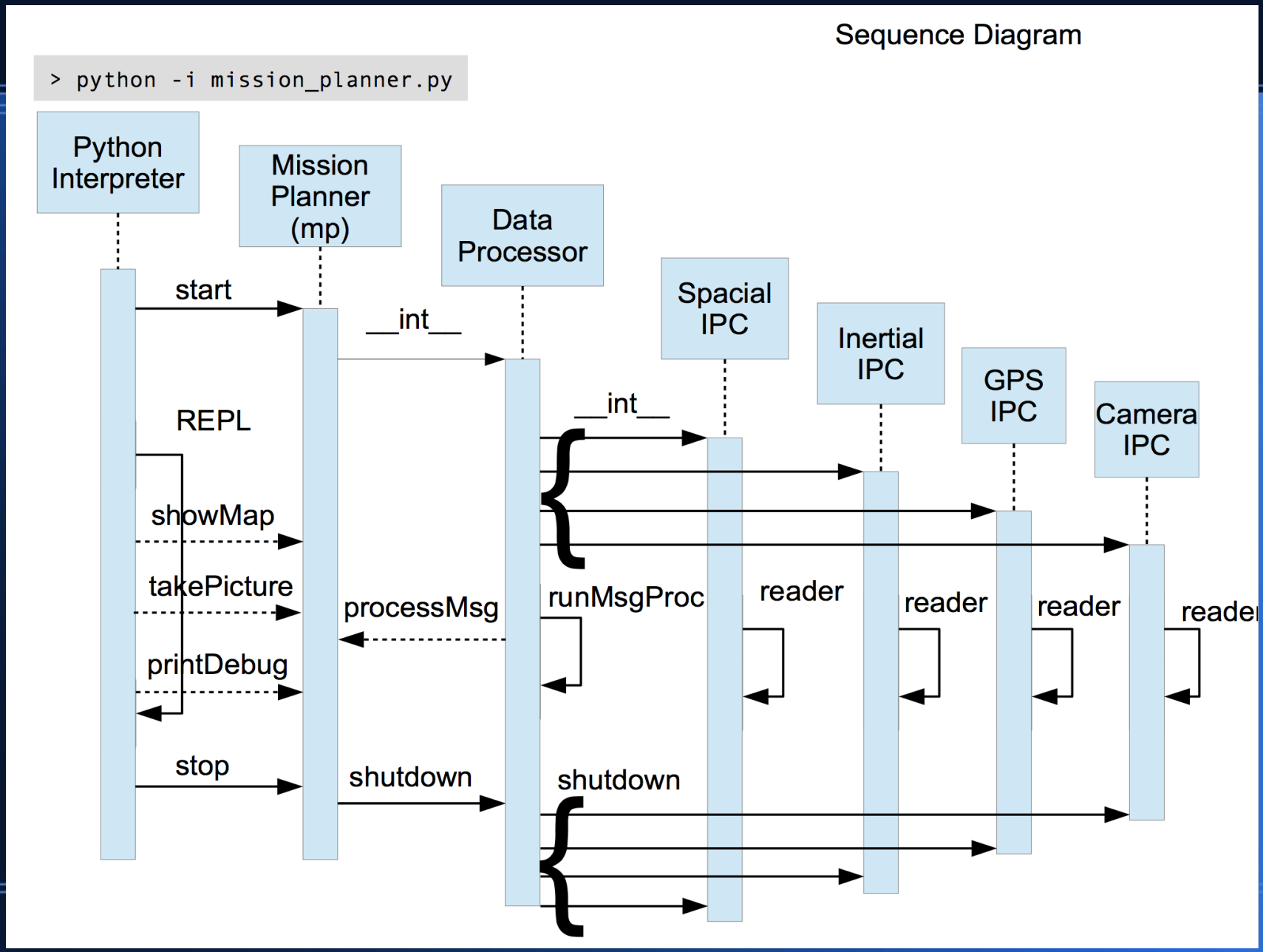
Software Architecture



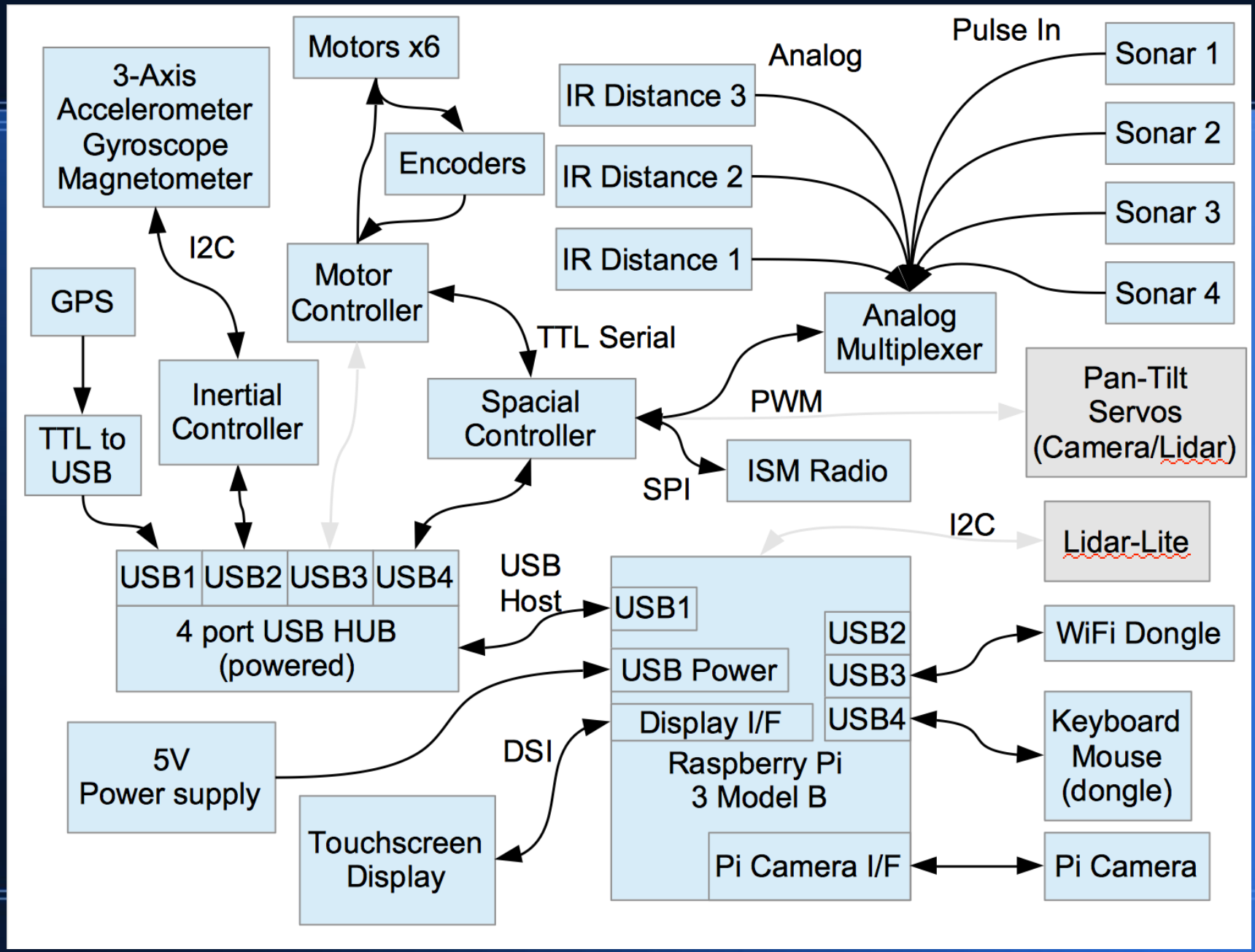
Software Arch. Continued



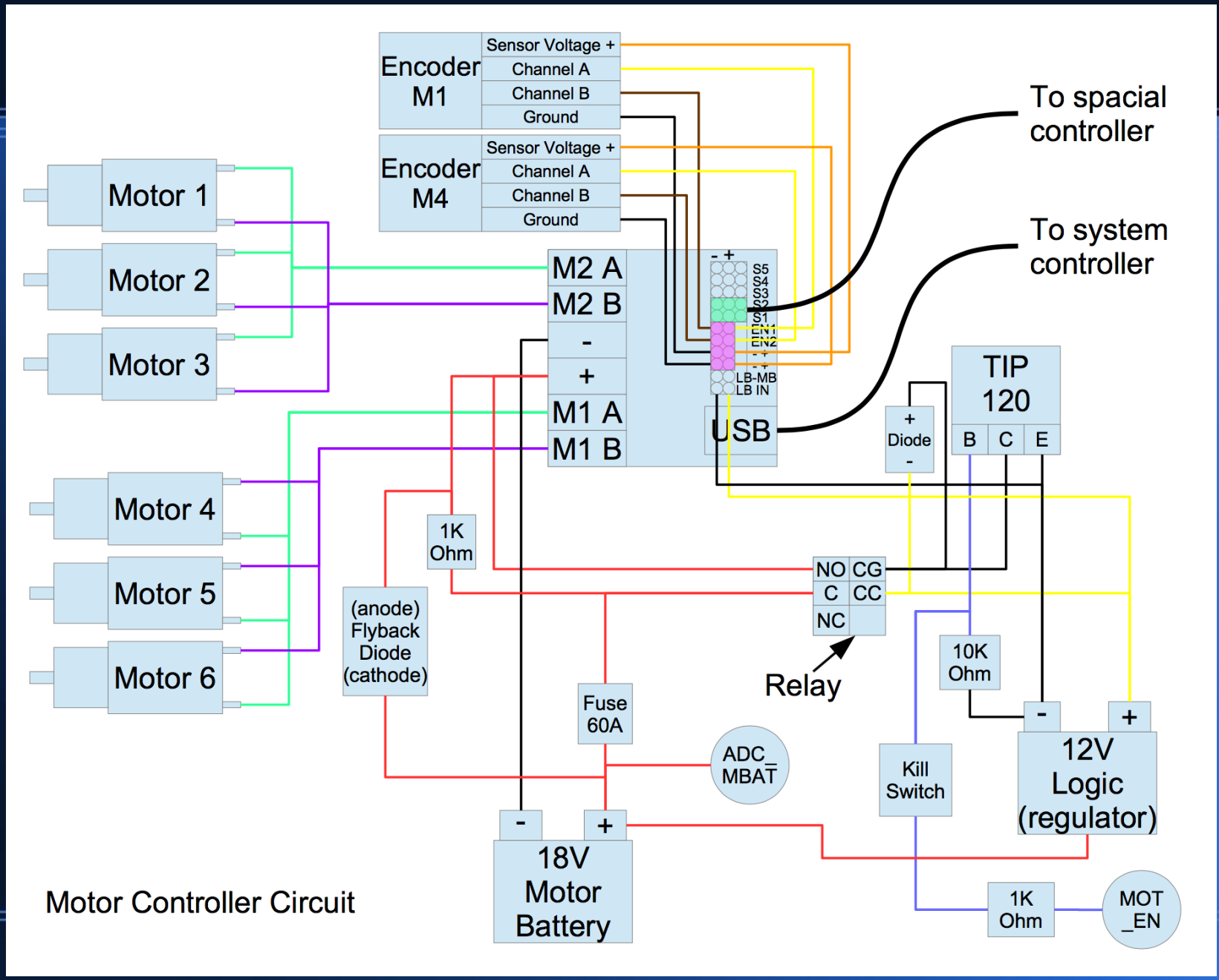
Software Arch. Continued...



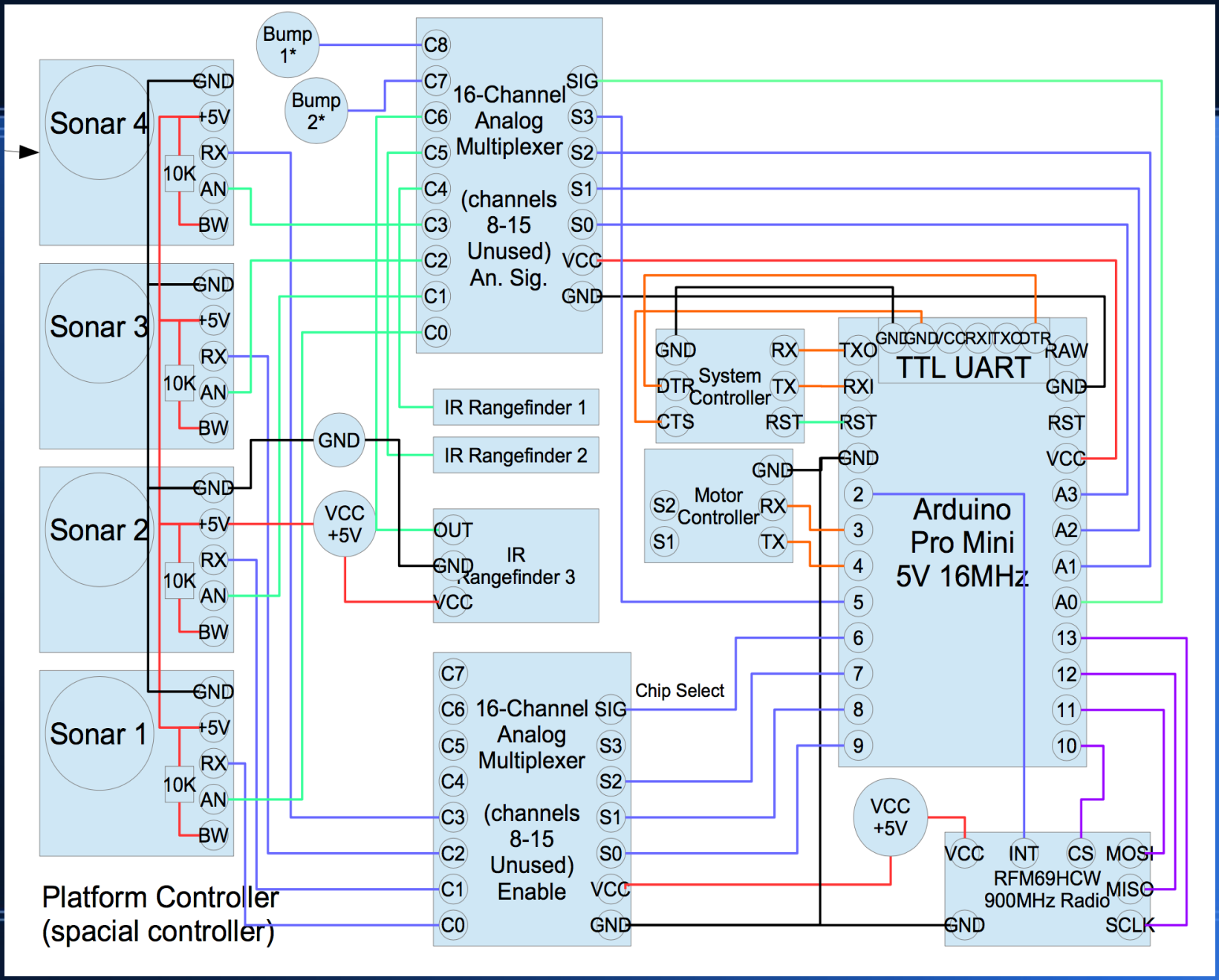
Hardware Block Diagrams



Electrical Diagrams/Schematics



Electrical Continued...



1st Competition, Lessons Learned

Bay Area Robogames 2017

- System/integrated testing is a must, SW is only 1 component
- Beware EMI/RFI from power electronics
- Read the data sheets...

2nd Competition, Was it Luck?

Seattle Robothon 2017

https://www.youtube.com/watch?v=bOybHa_b19g

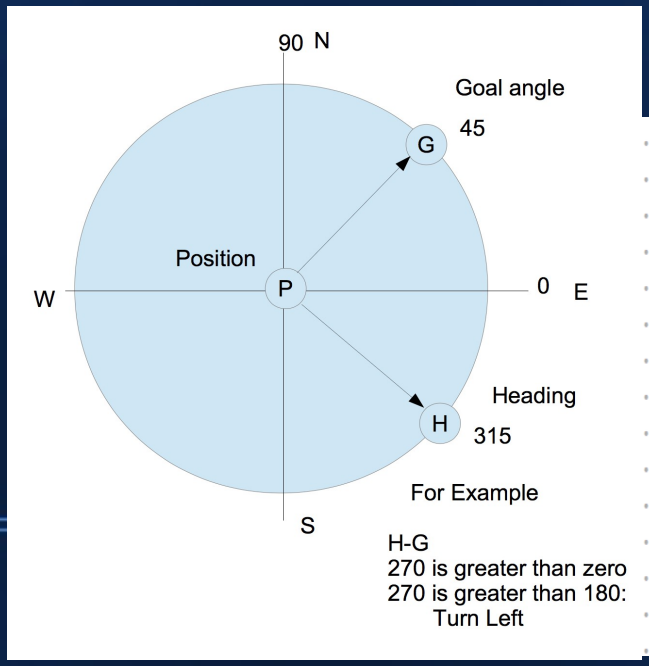
3rd Competition, 2nd in the world* :)

Bay Area Robogames 2018

https://youtu.be/wphYg5-G1YI/

Big Ideas, or at least ideas
(with respect to software and system design)

- Linux makes the robot's underlying software free, and changeable (if needed)
- Linux takes care of basic functionality, allows developers to focus on domain logic
- Ecosystem of programs and libraries, Arduino, OpenCV, etc.
- Lots of online help, Stack Overflow, Stack Exchange, Forums etc.
- Use libraries when you can: saves time, makes you a better SW dev.
- Onboard debugging and software update via TCP/IP, networking is native to Linux
- Python, an excellent language for rapid prototyping and development
- Arduino, an acceptable choice for an embedded controller...
- Testing is critical for success




```
# And determine whether we turn right or left.
HEADING_MINUS_ANGLE = heading - angle
if HEADING_MINUS_ANGLE < 0:
    if abs(HEADING_MINUS_ANGLE) < 180:
        self.__turnDirection = self.TURN_LEFT
    else:
        self.__turnDirection = self.TURN_RIGHT
elif HEADING_MINUS_ANGLE > 0:
    if abs(HEADING_MINUS_ANGLE) < 180:
        self.__turnDirection = self.TURN_RIGHT
    else:
        self.__turnDirection = self.TURN_LEFT
else:
    self.__turnDirection = self.TURN_STRAIGHT

```

References

- Seattle Robothon: https://robothon.org/
- Robo-magellan Rules: https://robothon.org/rules-robo-magellan/
- Bay Area Robogames: http://robogames.net/index.php
- Code on GitHub: https://github.com/EveRobotics/
- Seattle Robothon 2017: https://www.youtube.com/watch?v=bOybHa_b19g
- Robogames 2018: https://youtu.be/wphYg5-G1YI