Albatross: an open source UAV

Hugo Vincent
John Stowers

http://www.albatross-uav.org

ALBATROSS

What is a UAV?

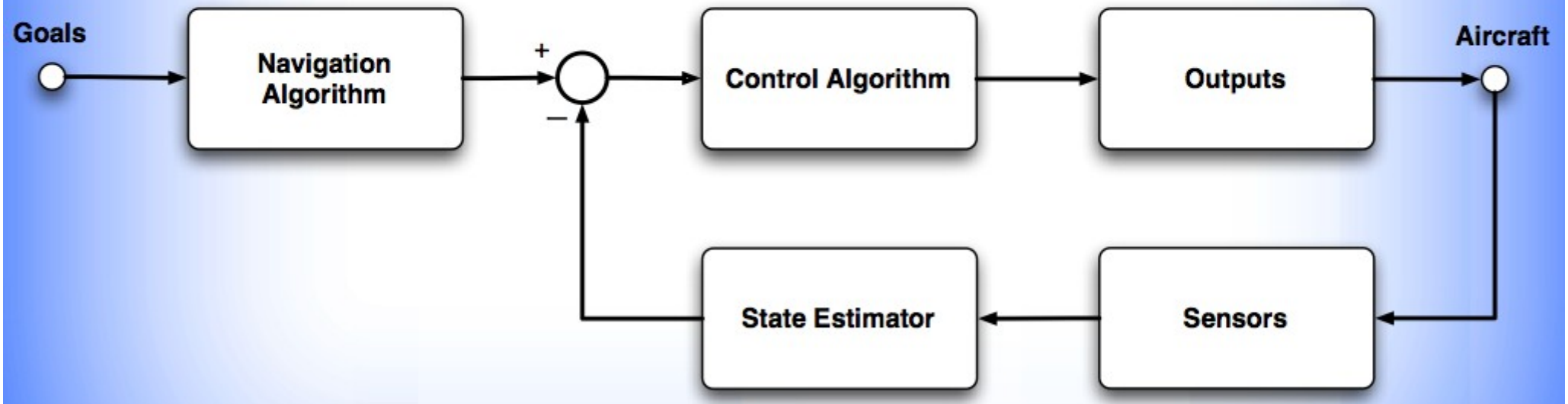
- Unpiloted Aerial Vehicle
- Computer replaces human pilot → autonomous flight
- Model aircraft airframe for devel.: slow, stable flight

Contents

- What we're doing
- Embedded Linux: controlling hardware, development environment, tool chain, etc ...
- Building hardware for Linux
- Open source aircraft
- The Future

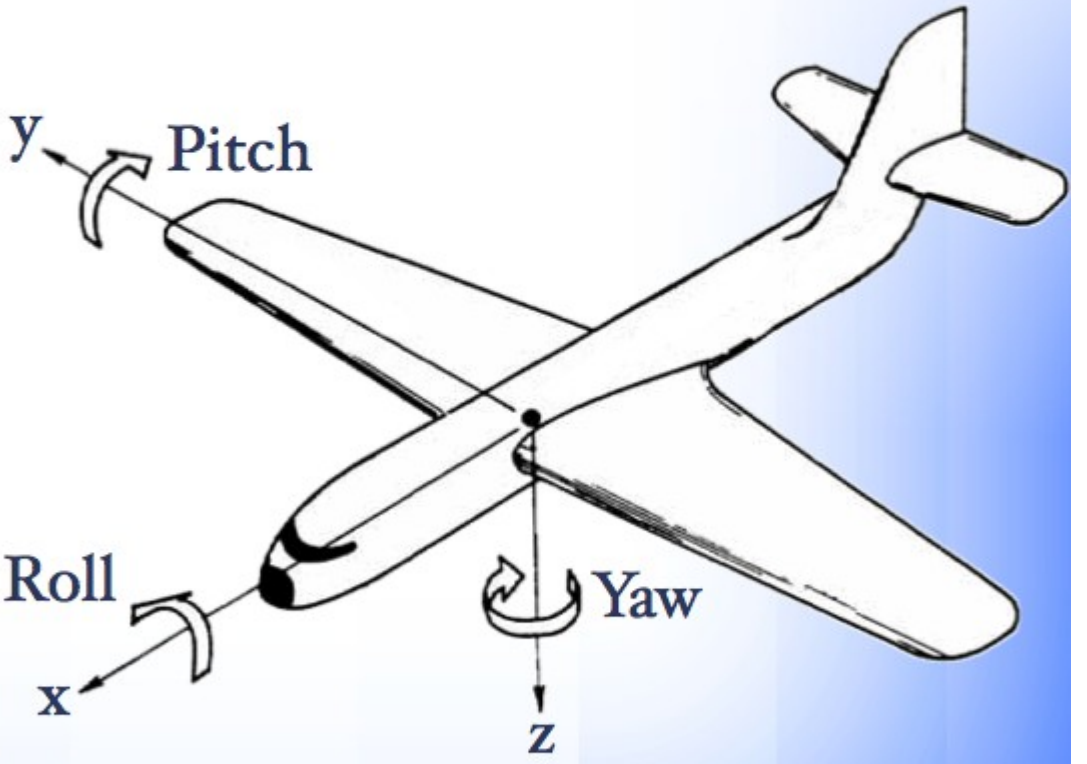
ALBATROSS

What it needs to do



Sensors

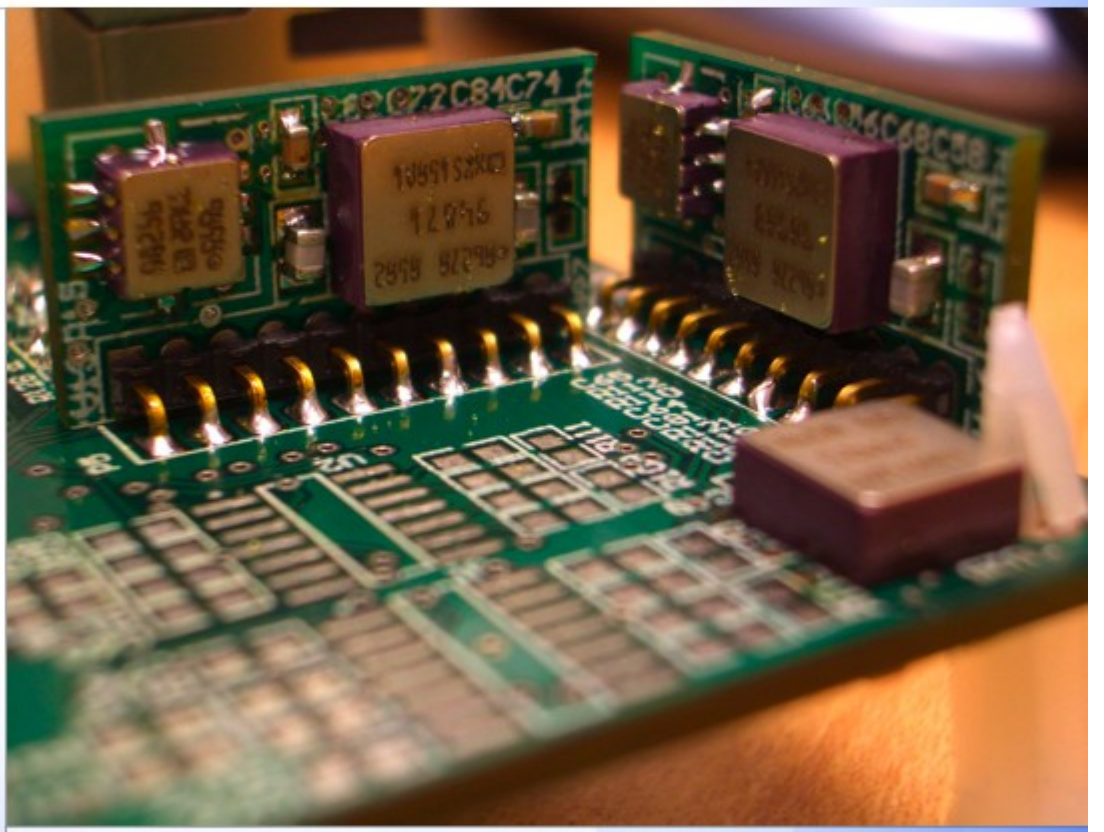
- GPS receiver
- Orientation needed for stabilization, but difficult to measure
- References:
  - Magnetic field (compass)
  - Gravity (accelerometer)



ALBATROSS

Inertial Sensors

- MEMS inertial sensors
- Linear accelerometers and rotation-rate gyroscopes
- Measure translation & rotation about 3 axis
- Data needs a lot of processing

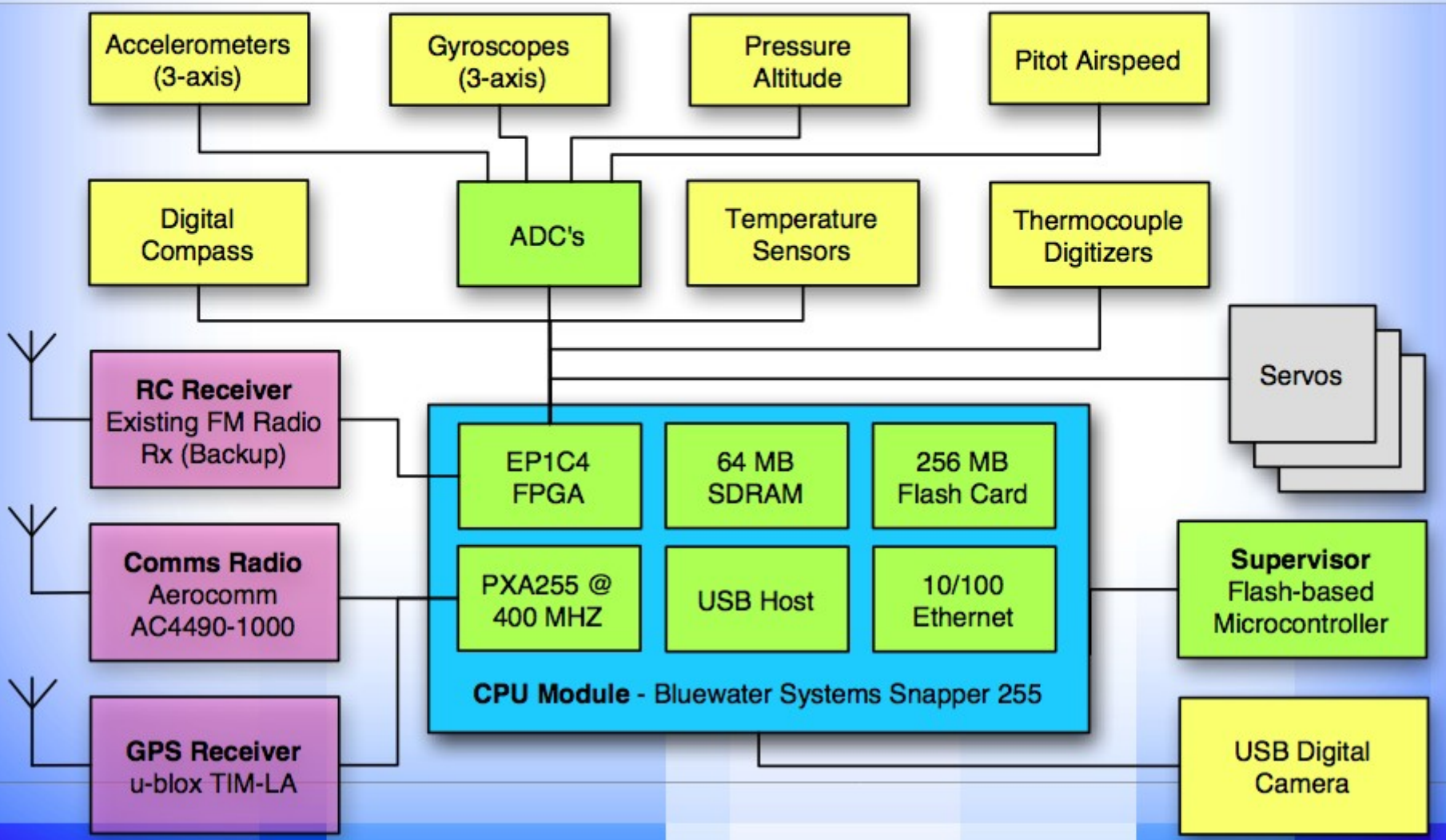


But... it's not that simple

- Sensor Drift
- Frames of reference vs. CPU time
- Noise & vibration: filtering
- Sample rate vs. latency
- Verification... and getting it down safely if things go wrong
- Control

ALBATROSS

Hardware



ALBATROSS

Hardware

- Start with commercial SBC, radio transceiver
- (Over -)Designed for expandability, flexibility
- 4 layer PCB, SMT

Software

- On Board - State, Control, Hardware, Comms, Logging
- Ground Station - monitor, command aircraft, plot flights
- Control: Five PID Loops

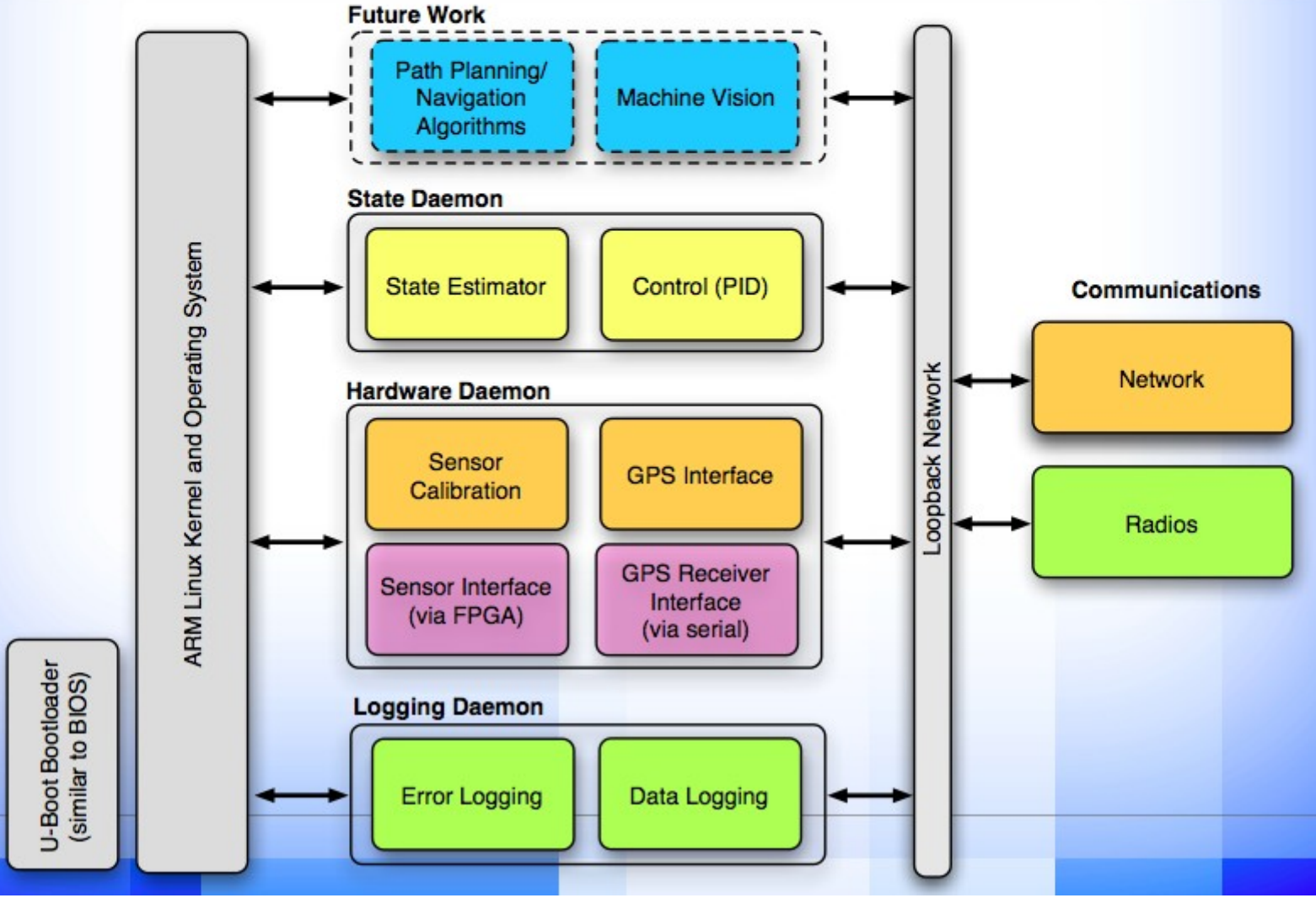
Development Rationale I: "Design For Test"

- How can we test without risking flying and crashing
- Verification - prove it works
- Simulation - demonstrate that it works
  - Different levels
- Software framework needed to make this easy

First try: Multicast UDP IPC

- Asynchronous, medium independent
- Each daemon broadcasts data and listens for data it needs on a shared bus
- No transmission control
  - Pros: Data should never be stale
  - Cons: Packets/data get lost

First try: Multicast UDP IPC



Second try: Abstraction through OO

- Single main loop (synchronous)
- Everything in one process
- Easier to deal with variable sample rates
- Use OO to get hardware/simulator independence

Development Rationale II: "Design For Rapid Development"

ALBATROSS

- Simulation
- Network & NFS Root
- Tool chains
- Accessing hardware on Linux
- Debug harnesses
- Languages & Structure

Hardware Problems

- No hardware FPU
- Memory bandwidth
- Asynchronous ADC sampling
- Unnecessary complexity due to SBC architecture
- Power supply over complexity
- Big, bulky, heavy, high power consumption

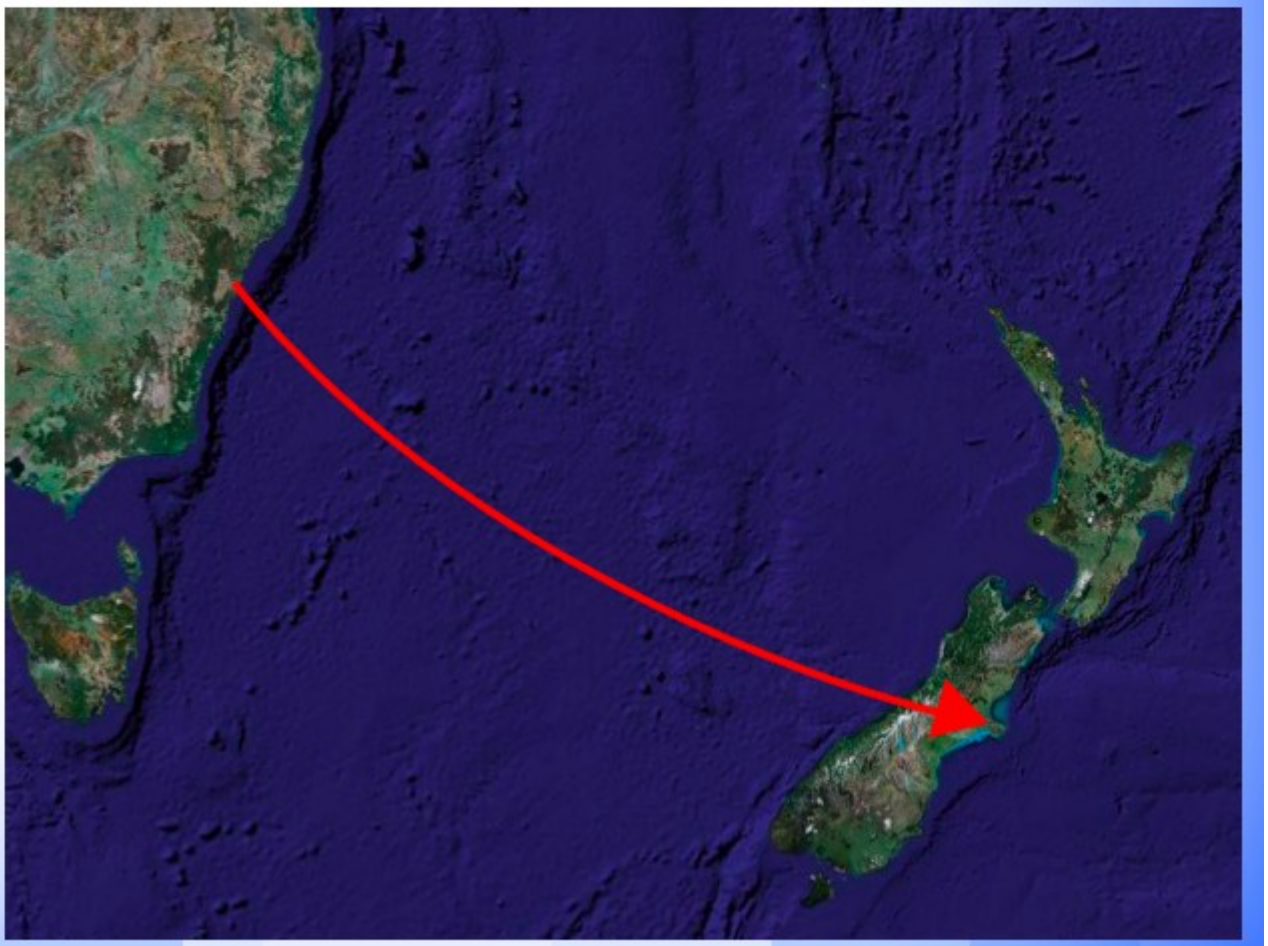
Hardware v2.0

- Custom design optimised for Linux and future goals
- CPU with HW floating point (and no more SBC, FPGA)
- Aircraft abstraction (power train management separated)
- Better sensors and calibration
- Much smaller and simpler

ALBATROSS

Future Goals

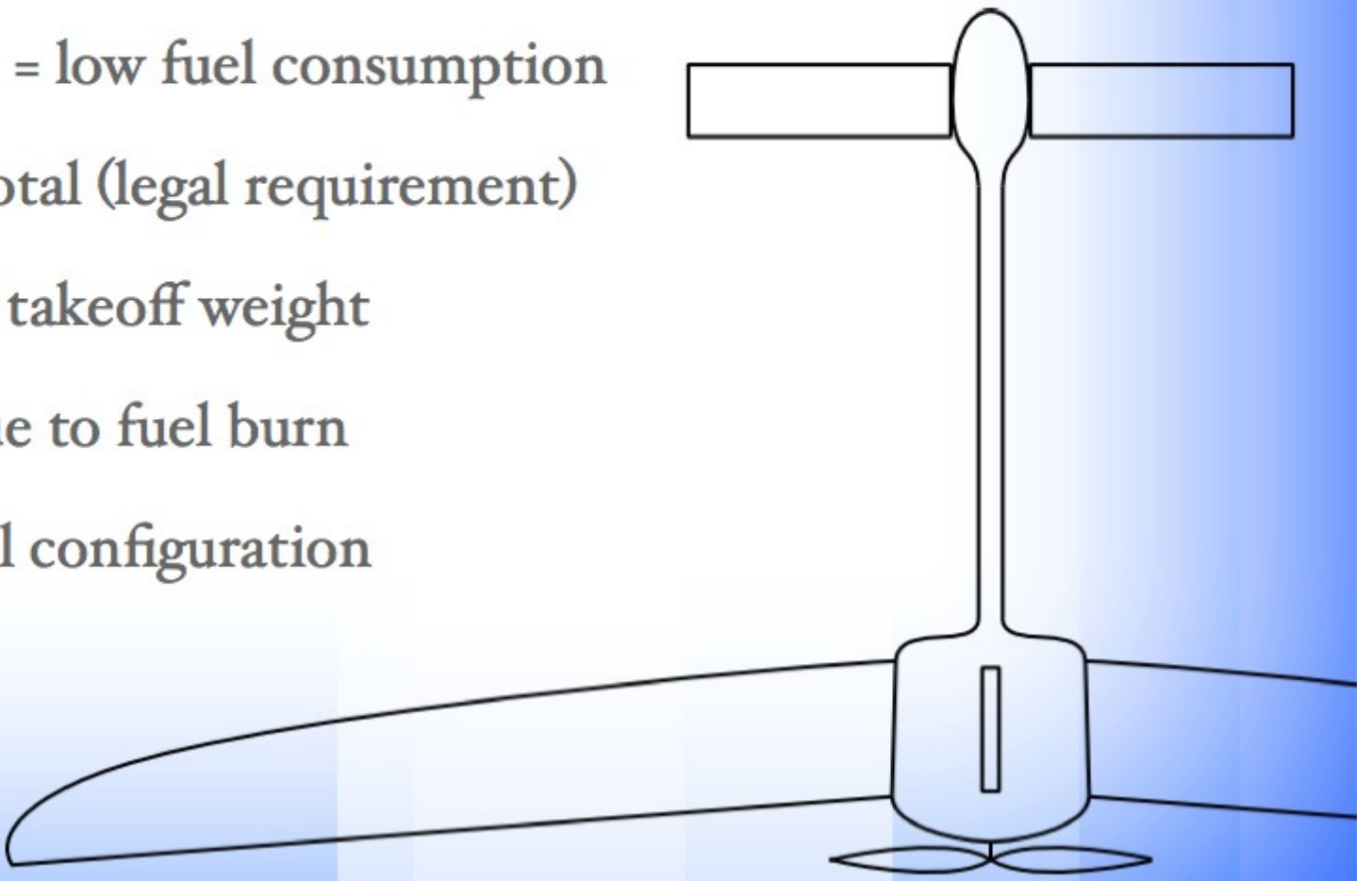
- Very long endurance



ALBATROSS

Aircraft Requirements

- Long endurance = low fuel consumption
- Less than 5 kg total (legal requirement)
- Fuel is 50% + of takeoff weight
- Varying CoG due to fuel burn
- Leads to unusual configuration



Engine Requirements

- 9 cc model aircraft engine
- The usual: small/light/cheap/reliable
- Fuel consumption at cruise speed: 70 - 90 mL / hour
- Continuous adjustment of fuel-air mixture
- Run reliably for 35 hours

Simulation Nirvana

- AVL (open source computational fluid dynamics)
- Control system design
- JSBSim flight simulation engine
- FlightGear flight replay and simulation

Questions?

ALBATROSS

State Estimation Algorithm