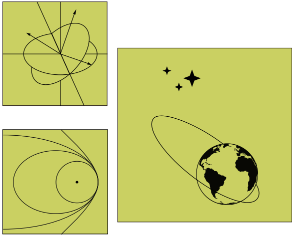
Open Source Rockets



Nathan Bergey
and
Andrew Greenberg

Portland State Aerospace Society

June 16, 2010

Portland State Aerospace Society (PSAS)

- A nonprofit, educational aerospace project at Portland State University started by students because introductory engineering labs *don't spectacular explode*.
- Consists of:
  - Undergraduate and graduate students at PSU.
  - Anyone interested in aerospace (e.g., you!).

What we want to do when we grow up?

Put a nanosatellite into orbit with our own rocket.

What we want to do when we grow up?

I mean really, how hard could that be?

Why Rockets

- Hard, system-level engineering design problem.
  - Intelligently resolving problems, better.
  - Microelectronics and computational horsepower change the game.
- Nerd Sniping.
- Science!
  - Atmospheric, aurora, X-ray and infra-red astronomy, etc.
- Space is the future...eventually.
- **They're just really cool.**

Rocket Science is, actually, hard

It's not necessarily complex, but it's very hard to get right.

The Rocket Gap

Tons of people build model rockets.

Figure: Not Quite Cape Canaveral CC-BY Unhindered by Talent

The Rocket Gap

Lots of people build High Powered Rockets (HPR).

Figure: CC-BY jurvetson

The Rocket Gap

Some HPRs have even reached space.

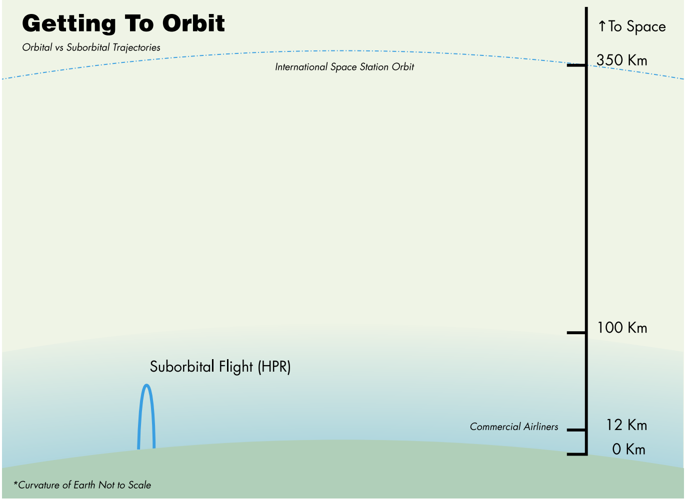
Figure: CSXT/GoFast space launch, May 17, 2004. CC-BY-SA Ian Kluft KO6YQ

The Rocket Gap

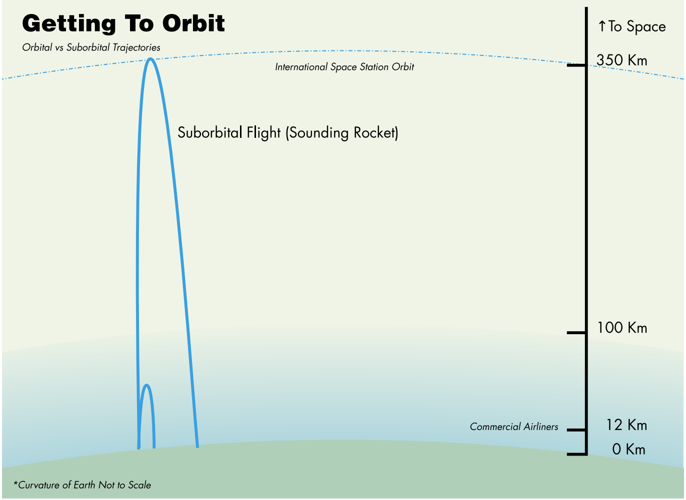
But none are capable of ever reaching orbit.

Figure: DRAGONSat after its release during STS-127. Credit: NASA

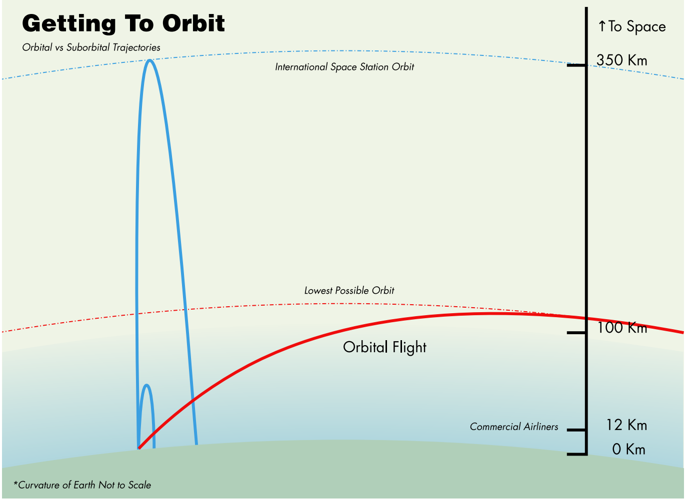
What is Orbit, anyway?



What is Orbit, anyway?

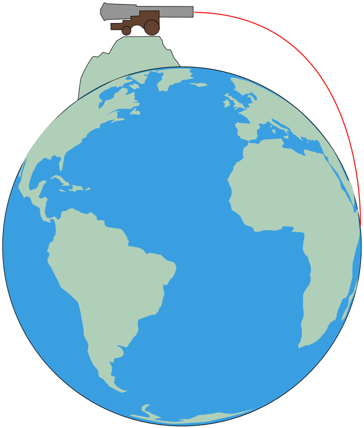


What is Orbit, anyway?

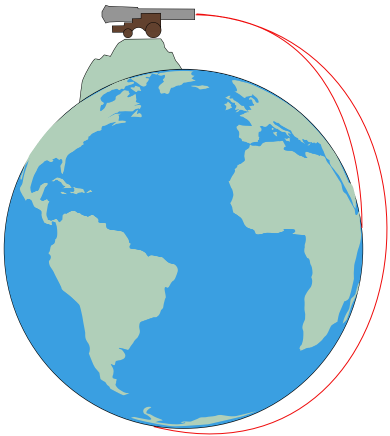


Newton's thought experiment: A big cannon.

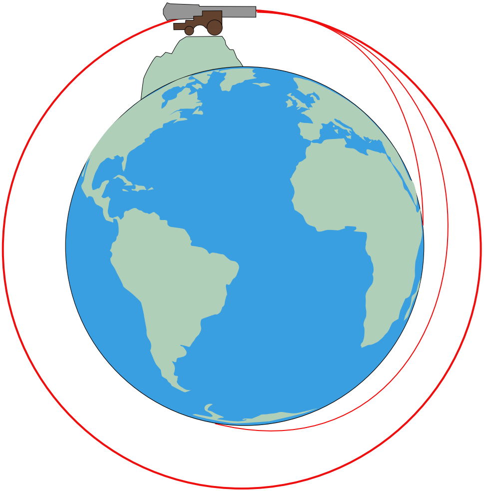
Newton's thought experiment: A big cannon.



Newton's thought experiment: A big cannon.

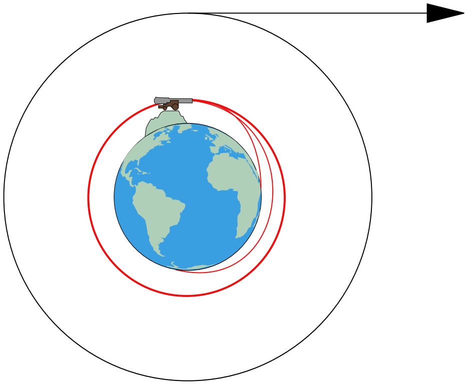


Newton's thought experiment: A big cannon.

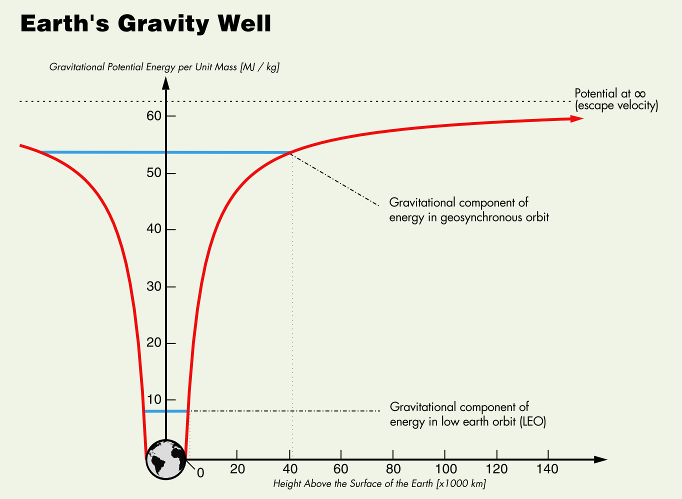


Just how fast sideways?

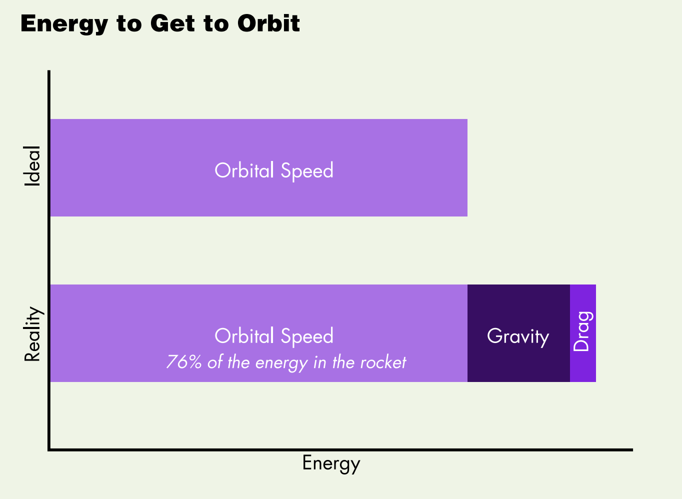
17,500 mph (Mach 23)



And we're stuck in a well!



Where the energy goes



Some comparisons of orbital energy

- Taking 1 kg to orbit requires about 32.4 megajoules of energy.
- That's the same as:
  - The energy in a freight train going 34 MPH.
  - The energy in 155 pounds of TNT.

Small and lightweight: Nanosatellites

Figure: In-flight picture of CP4 taken by AeroCube2, ©The Aerospace Corp. And Cal Poly student Marissa Brummitt displays a CubeSat at a Central Coast Astronomical Meeting. CC-BY Waifer X

Recap: what you need to get to orbit

- Big (!) rocket to get into space and acheive orbital velocity.
- Oh, right: and lawyers. Lots and lots of lawyers.
- Some way to **steer** the rocket to get it into an orbit.

Rocket steering

Autonomous steering (of anything) requires 3 things:

- **G**uidance - Where am I going?
- **N**avigation - Where am I now?
- **C**ontrol - How do I get from here to there?

Guidance: Where are we going, and why are we in a handbasket?

- The trajectory (path) from ground to orbit.
- Plain old vanilla rocket science:
  - Figure out the least energy solution to get you to orbit.
  - 25 cent phrase: "optimal orbital insertion trajectory".
- Mostly calculated in advance, depends on:
  - Launch site location.
  - Orbit you're trying to acheive.
  - How your rocket works.
  - The current atmospheric conditions ("weather")

Navigation: where are we really?

- More precisely, what is the full inertial state of the rocket?
  - position, velocity, acceleration
  - attitude, rotational velocity, rotational acceleration
- That's easy to figure out, right?

Navigation: No, this is the *hard* part

- Knowing your inertial state accurately is really, really, hard.
- *This* is the problem we're most interested in trying to solve.

Navigation: IMU

Inertial Measurment Unit

- 6 degrees of freedom (6 DOF) of movement
  - 3 linear accelerometers (X, Y, Z)
  - 3 rotational gyroscopes (roll, pitch, yaw)
- But really **18** variables: x, $\dot{x}$, $\ddot{x}$, y, $\dot{y}$, $\ddot{y}$, z...

- Comercial strategic grade IMU's - Forget it.
- Comercial tactical grade IMU's - $250,000.
- Roll your own extremely crappy IMU! Only $200!!
  - Crappy = drifts over time (integrates noise like t^2)
  - Position accuracy is gone within 10's of seconds.

Navigation: GPS

Global **P**ositioning **S**ystem

- Provides relatively slow absolute position updates.
- Doesn't provide attitude
- Could lose signal and/or signal lock.

Navigation: Other sensors

- Magnetometers: 3D compasses, gives attitude but not position.
- Barometric altimeters: gives height above ground only.
- Image processing: Gives attitude, but probably not position.

Navigation: Each of these sensors... well... they suck.

- No way to robustly get inertial state using individual sensors.
- But what if you could *combine all* of the sensors?

Navigation: Data fusion, the magic sauce

- Data fusion techniques allow us to optimally combine navigation sensors
  - Absolute position of GPS corrects for IMU drift.
  - Inertial state from IMU aids GPS correlator lock.
  - Magneometer attitude aid gyroscope drift.
  - barometric altimeter aid GPS and inertial height estimates.
- Example data fusion techniques: Kalman filtering, particle filters, etc.
- Currently using Bayesian Particle Filters

Control: How do you steer this thing?

- People with actual resources use thrust vector control (TVC).
  - Gymballed nozzles, like on the space shuttle.
- Poeple without so many resources use more crude methods:
  - In the lower atmosphere, how about fins?
  - In space, how about cold gas jets?

Control: Who's got the steering wheel?

- Control "Closes the loop" from navigation to guidance: from where we are to where we want to go.
- Inertial and aerodynamic models help predict what to do (e.g., model preditive control).

What Have we Done So Far?

- Started in 1997
- Taken many small steps
- Had our share of failures along with some big successes

Launch Vehicles

- LV0
- LV1
- LV2.1
- LV2.2
- Current work: LV2.3

Launches

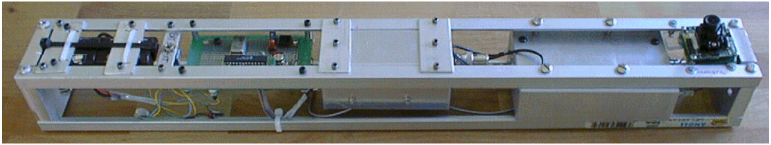
Date	Vehicle	Altitude	Status
1998/06/06	LV0	0.3 km (1,000 ft) AGL	Success
1999/04/11	LV1	3.6 km (12,000 ft) AGL	Success
2000/10/07	LV1b	3.6 km (12,000 ft) AGL	Success
2002/09/22	LV2.1	18,848 ft AGL	Success (airframe only)
2003/08/23	LV2.2	N/A	Failure (airframe only)
2005/08/20	LV2.1	18,805 ft AGL	Failure (avionics success)
2009/05/31	LV2.3	12,600 ft AGL	Success (airframe only)

1998 - Launch Vehicle No. 0 (LV0)

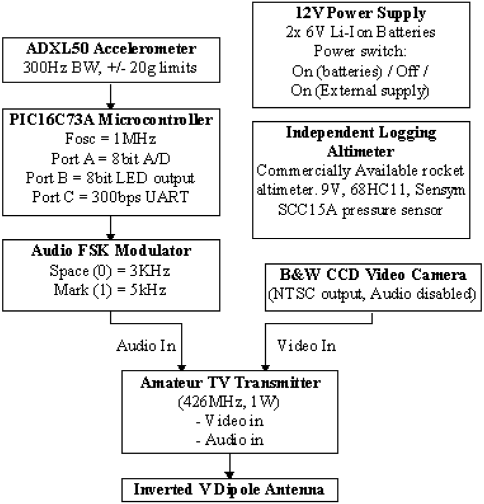
- PSAS “first attempt” at high power rocketry
- Prototype an amateur television-based telemetry system
- Flight computer: 8 bit RISC microcontroller (PIC 16C73A)
- Typical “hack” project
- Apogee: 0.3 km (1,000ft)

LV0

LV0 Avionics



LV0 Avionics



LV0 Ground Control

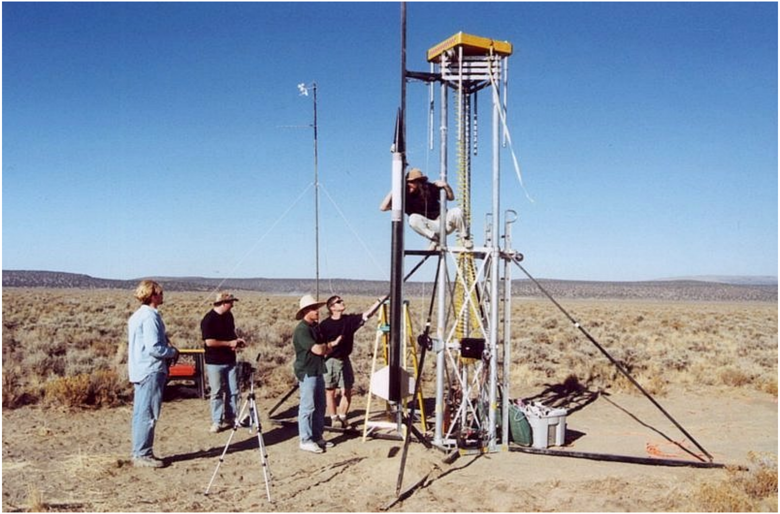
LV0 Lessons

- We could actually build a rocket that did something.
- Amateur radio clearly the way to go.

2001 - Launch Vehicle No. 1 (LV1)

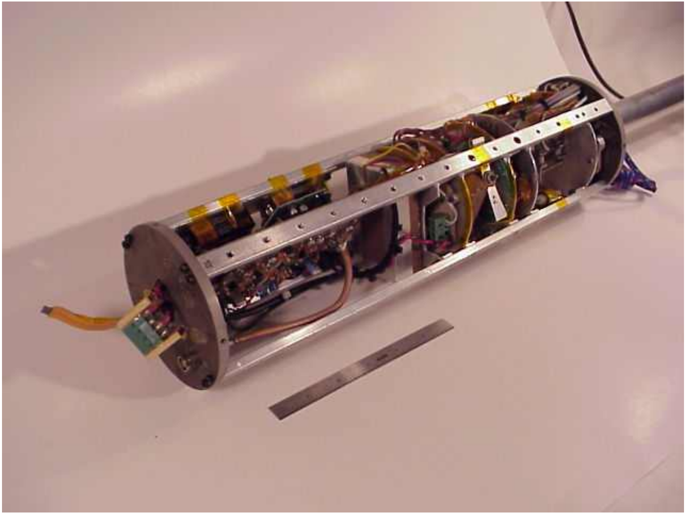
- Proof of concept of airframe design
  - 3.35m (132 in), 11.1cm (4.37 in) OD, 19.5kg (43 lb) CF/FG body
  - 7755 Ns solid proprellant motor ("M")
- Proof of concept avionics system
  - Inertial Measurement Unit (IMU) and GPS
  - High speed telemetry system with emergency uplink
- Apogee: 3.6 km (12,000ft) AGL

LV1

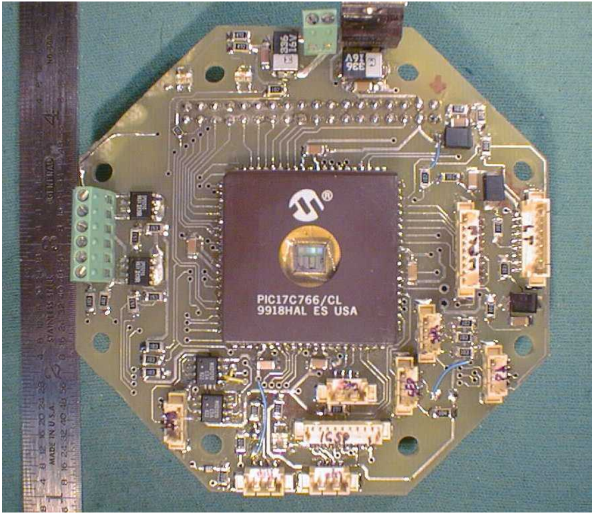


LV1

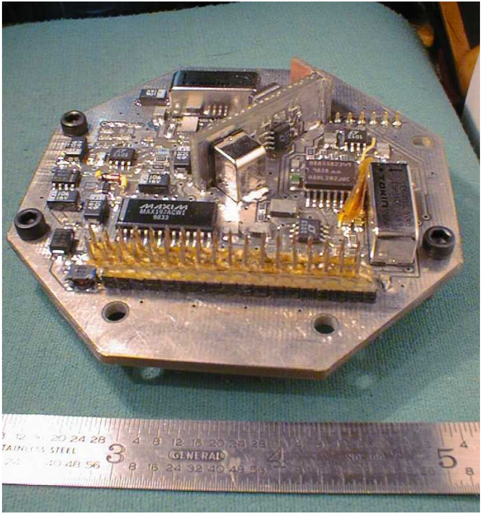
LV1b avionics system



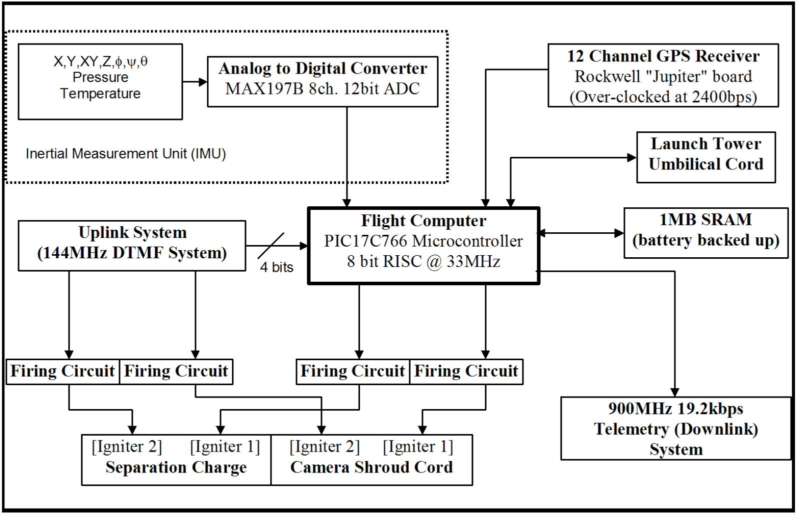
LV1b flight computer



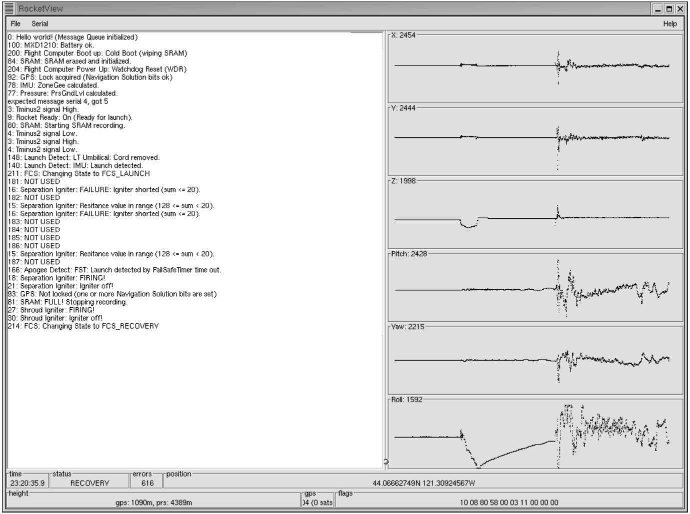
LV1b IMU



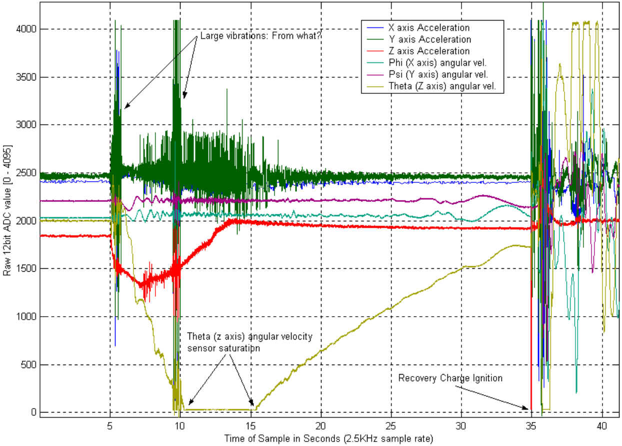
LV1b avionics



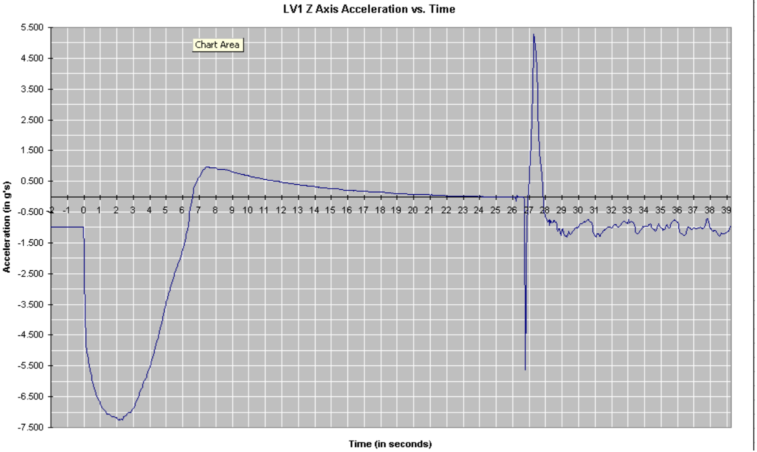
LV1b ground software



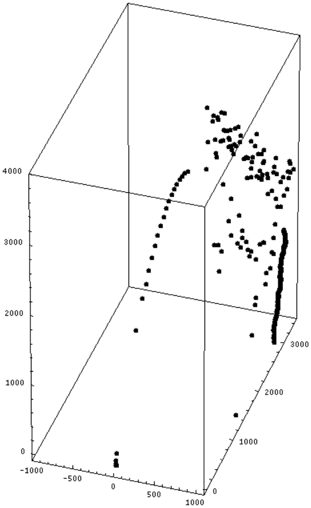
LV1b data - Raw IMU data



LV1b data - Z axis acceleration



LV1b data - GPS



LV1 Flight Videos

- 1999-04-11 ground video
- 1999-04-11 on-board video

LV1/1b Lessons

- Need much more powerful flight computer.
- Need more thought into system design (e.g., batteries).
- Better collaboration tools needed.

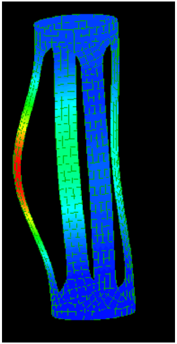
2005 - Launch Vehicle No. 2.1 (LV2.1)

- Designed from the ground up as a modular test bed for inertial navigation
  - 4.02 m (13.2 ft), 13.6 cm (5.37 in) OD, 27 kg (60 lbs) Al and FG body
  - Designed for “N” and “P” motors.
- Apogee: 5.5 km (18,000ft) AGL
  - Theoretical 23 km (75,000ft) AGL on “P”

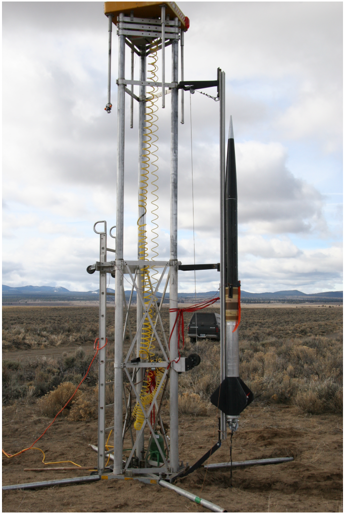
LV2.1 airframe

- 5.25 in O.D. x 1/8 in wall Aluminum modules
- 0.06 in thick aeroshell made from vacuum formed 8 layer Eglass/epoxy sandwich
- 7:1 Parker nose cone (optimized for Mach 3-4)

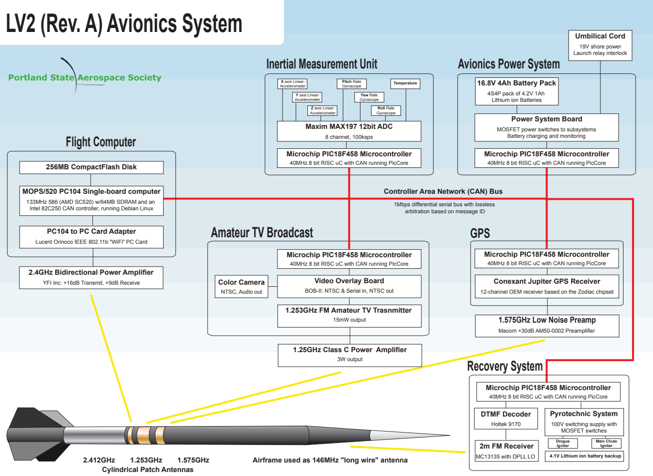
LV2.1 airframe



LV2.1 airframe



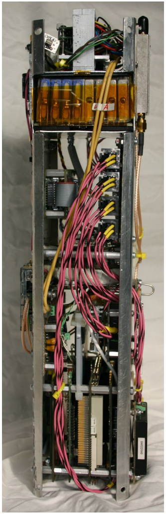
LV2.1 Avionics System

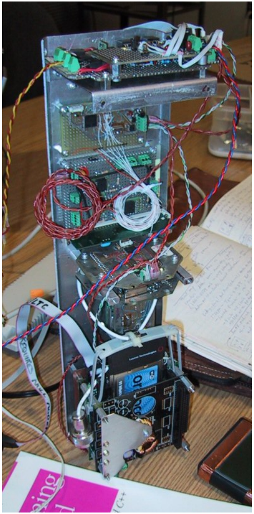


LV2.1 on-board software

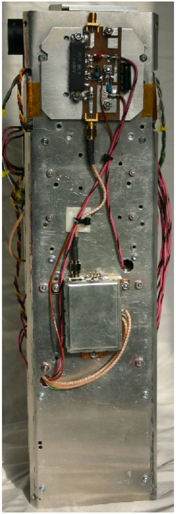
- Debian Linux with 2.4.27 kernel
- ADEOS patch enabled real time interrupts on CAN driver
- Embedded Debian by stripping out unnecessary software
  - No: X, cron, syslog, docs, development tools, etc.
  - Kept: ssh, bash, vi, wireless-tools, etc.

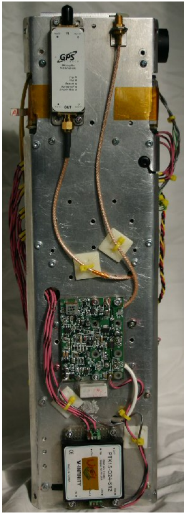
LV2.1 avionics system



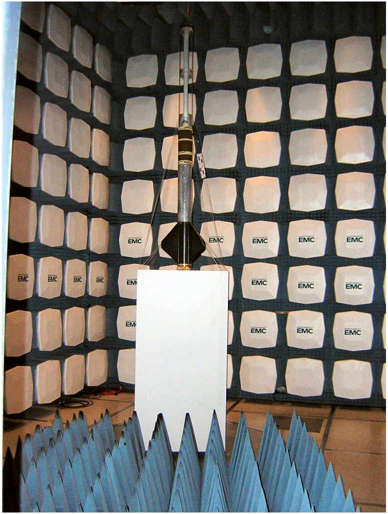


LV2.1 communication system



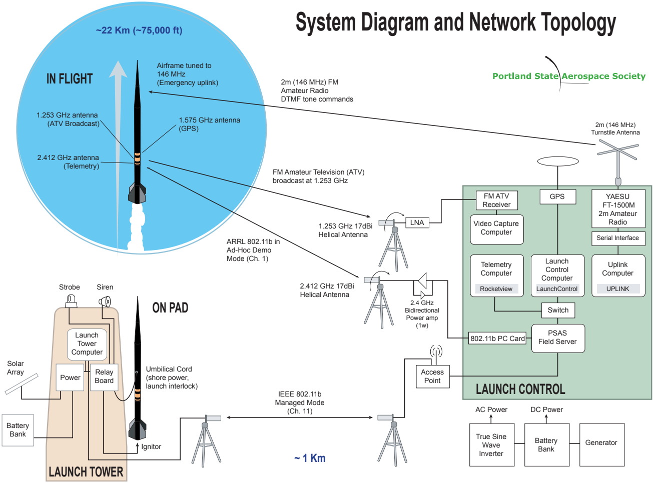


LV2.1 cylindrical patch antenna testing



LV2.1 ground system communication

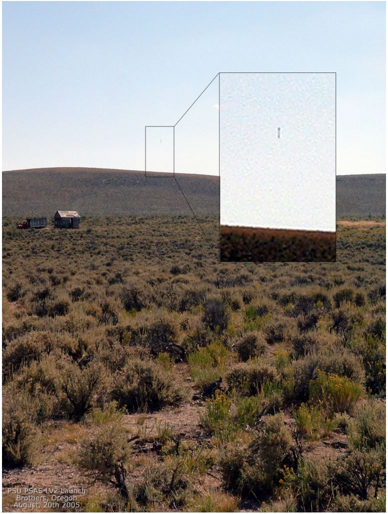
LV2.1 System



System Diagram and Network Topology
Portland State Aerospace Society
~22 Km (~75,000 ft)
IN FLIGHT
Airframe tuned to 146 MHz (Emergency uplink)
1.253 GHz antenna (ATV Broadcast)
1.575 GHz antenna (GPS)
2.412 GHz antenna (Telemetry)
2m (146 MHz) FM Amateur Radio DTMF tone commands
2m (146 MHz) Turnstile Antenna
FM Amateur Television (ATV) broadcast at 1.253 GHz
1.253 GHz 17dBi Helical Antenna
LNA
ARRL 802.11b in Ad-Hoc Demo Mode (Ch. 1)
2.412 GHz 17dBi Helical Antenna
2.4 GHz Bidirectional Power amp (1w)
FM ATV Receiver
GPS
YAESU FT-1500M 2m Amateur Radio
Video Capture Computer
Serial Interface
Telemetry Computer
Rocketview
Launch Control Computer
LaunchControl
Uplink Computer
UPLINK
Switch
802.11b PC Card
PSAS Field Server
Access Point
LAUNCH CONTROL
AC Power
DC Power
True Sine Wave Inverter
Battery Bank
Generator
Strobe
Siren
ON PAD
Launch Tower Computer
Solar Array
Power
Relay Board
Umbilical Cord (shore power, launch interlock)
Battery Bank
Ignitor
LAUNCH TOWER
IEEE 802.11b Managed Mode (Ch. 11)
~ 1 Km

LV2.1 launch control

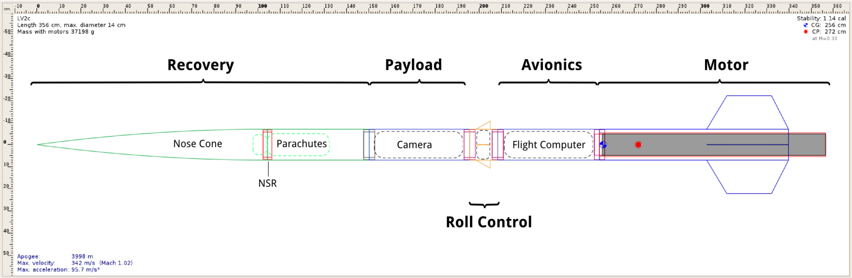
LV2.1 "lawn dart"



LV2 Lessons

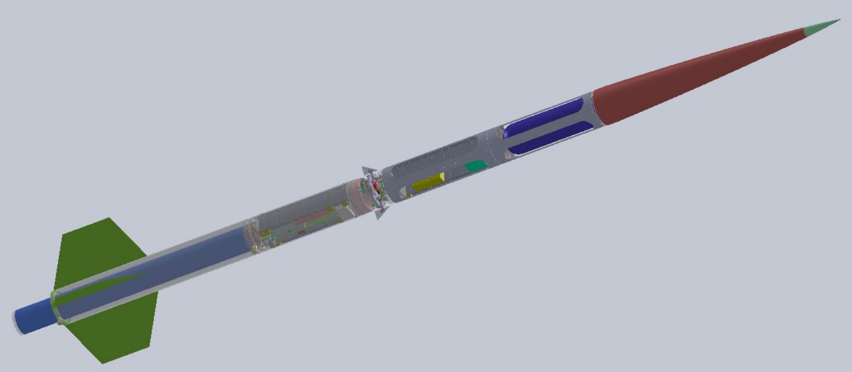
- Up-front design really works.
- Total avionics success: WiFi at $>$ Mach 1 !
- Old gunpowder at 22,000 ft doesn't work.

LV2.3 plan

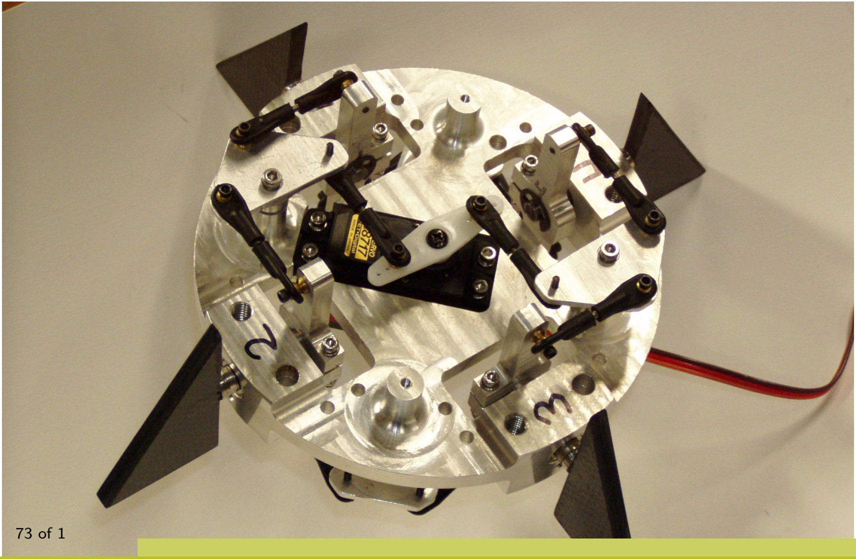


LV2c
Length 356 cm, max. diameter 14 cm
Mass with motors 37198 g
Stability: 1.14 cal
CG: 256 cm
CP: 272 cm
at M=0.30
Recovery
Payload
Avionics
Motor
Nose Cone
Parachutes
Camera
Flight Computer
NSR
Roll Control
Apogee: 3998 m
Max. velocity: 342 m/s (Mach 1.02)
Max. acceleration: 95.7 m/s²

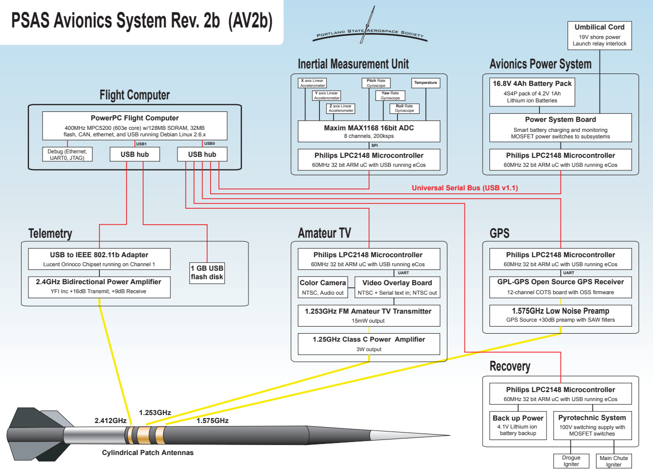
LV2.3 CAD airframe



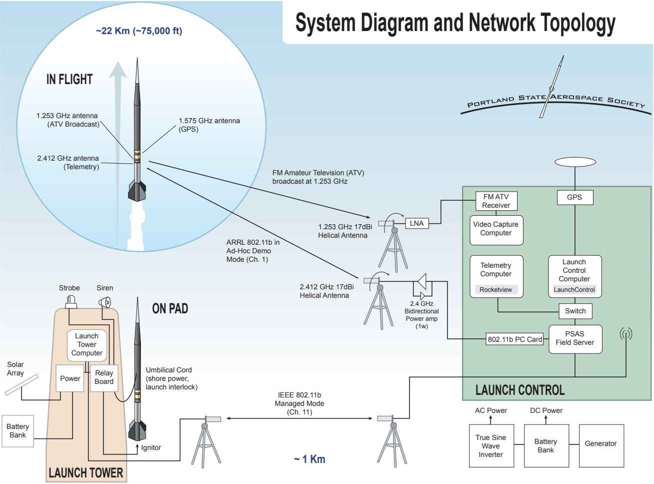
LV2.3 Roll Control



AV3 avionics system



LV2.3/AV3 System



LV2.3 on tower

LV2.3 launch

LV2.3 Flight Video

- 2005-05-31 high speed (600 fps) ground video

Finally

So, uh, maybe we should talk about open source, huh?

Standard collaboration tools

- Debian stable Linux server
- wiki: **ikiwiki**
- version control: **git**
- mailing lists: **mailman**
- Desktop tools:
  - **LaTeX (beamer)**
  - **OpenOffice**
- Visualization:
  - **gnuplot**
  - **inkscape**
  - python's **matplotlib**
- Languages: mostly C and Java, some Python.
- Pretty standard stuff, except perhaps ikiwiki.

Open software tool chains

- The rocket is an example of an embeded system. We use common tools for writing and debuging code on the flight hardware.
  - gcc
  - gdb
  - make
  - eclipse (w/CDT)
  - openocd (JTAG driver)
- We choose hardware for which open tools exist.
  - ARM
  - PowerPC
  - x86
  - **not** PIC
- We use Linux as an embedded OS on big chips (x86, PPC).
- We use FreeRTOS (or go "bare metal") on everything else.
- Most ground system PC-based software is written in Java for cross-platform use.

Open Source CAD: Mechanical

- 2D drawing and layout
  - QCAD: Works great, albeit with a clunky UI.
- 3D drawing and solid modeling
  - *Critical* for complicate electro-mechanical assemblies
  - FreeCAD: 3D file viewer, can't yet edit and design files. We have high hopes for this one.
  - Current stopgap: we use SolidWorks under an academic license.
- Computational Fluid Dynamics
  - OpenFOAM: 3D , very difficult to use, might even work.

Open Source CAD: Electrical

- Schematic capture and PCB layout
  - gEDA: collection of cobbled together tools. As of the latest release, usable! Ongoing integration work will make it much better.
  - KiCAD: much more polished than gEDA, missing one or two bits of critical functionality (e.g., “undo” in the PCB editor).
  - Current stopgap: Using EAGLE CAD under the free license.

Open Source CAD: Rockets

- Model and high power rocket simulation
  - OpenRocket: a fantastic cross-platform replacement to "RockSim", allows design and simulation of high power rockets. A clear win for open source in rockets.
- Orbital mechanics
  - OTIS: NASA's Orbital trajectory by implicit simulation. Under ITAR, unfortunately.
  - In the process of rolling our own rocket simulator.

Open Standards

We try to use standards based technology wherever we can.

- 802.11
- ARRL standards (ham radio)
- Standards based buses:
  - Controller Area Network (CAN)
  - USB
  - SMB
  - JTAG

Open Source hardware

- What does “open hardware” really mean?
  - Requires enough info to replicate hardware—whatever that means.
  - CAD documents: mostly they’re incompatible across CAD programs.
  - “Standards“ based source files not useful: text, PS/PDF, Gerber..
  - Bundled firmware and software (under GPL?)

Open Source hardware

- What license to use?
  - GPL doesn't really address hardware issues.
  - TAPR Open Hardware License handles hardware, but overbearing.
  - Others?

Open Source hardware

- Open hardware issues
  - There are a **lot** less open hardware folks than open software folks
  - More expensive and it sucks when you fab other people's mistakes.
  - Open source software revolves around shared tools, hardware tools don't yet have the same large and adopted ecosystems.
  - Hardware (usually) requiers a lot more investment to get something to work, not as easy to give away.
  - Open source is established as legitimate buisness model (we think?), but open hardware models are not yet tested. All eyes on "Chumby"?

Open source issues

- We will soon have to deal with the International Traffic on Arms Regulations (ITAR).
  - Remember the PGP debacle? We don't want to, either.
  - Protecting ourselves from legal action while pursuing orbit will be difficult: will have to ask for outside legal aid.
- Like many complicated open projects, we have serious documentation organization and management issues.
  - How do you organize thousands of documents? That span over time? And multiple projects?
  - How do you find things quickly when search fails you? We use the dreaded phrase "it's on the wiki" quite often.
- Upgrading tool chains is always hard:
  - CVS to SVN to git (fairly painless).
  - Twiki to MoinMoin to ikiwiki (very, very painful).

Open source wins

- PSAS would not be possible without open source tools: our current project just wouldn't have been possible 10 years ago.
- New, better OSS tools let us increase productivity (git, ikiwiki, eclipse, etc).
- "If we just wait another 6 months..."

Portland State Aerospace Society (PSAS)

- Thank you! Any questions?
- http://psas.pdx.edu/
- Follow @pdxaerospace
- Or follow me (Nathan) @natronics