Open Source Software for Commercial Off-the-Shelf GPS Receivers

Andrew Greenberg

Portland State Aerospace Society

Takuji Ebinuma

Mitsubishi Electric Corporation, Kamakura Works

Overview

- Introduction to Open Source
- Motivations behind GPL-GPS
- Choosing a Chipset and Receiver
- Choosing a RTOS
- GPL-GPS: Take OSGP, add eCos, stir well.
- Status and Initial Results
- Future work

Introduction to Open Source and Open Source GNSS Projects

Open Source Software

- Not just software that comes with source code.

- Two versions of open source:

  1. It's yours, do whatever you want with it.
     (Berkeley License, or **BSD**)

  3. It's yours, do whatever you want with it privately. But *distributions* of the software, *and modified versions of the software*, must also be open source.
     (Free Software Foundation's General Public License, or **GPL**)

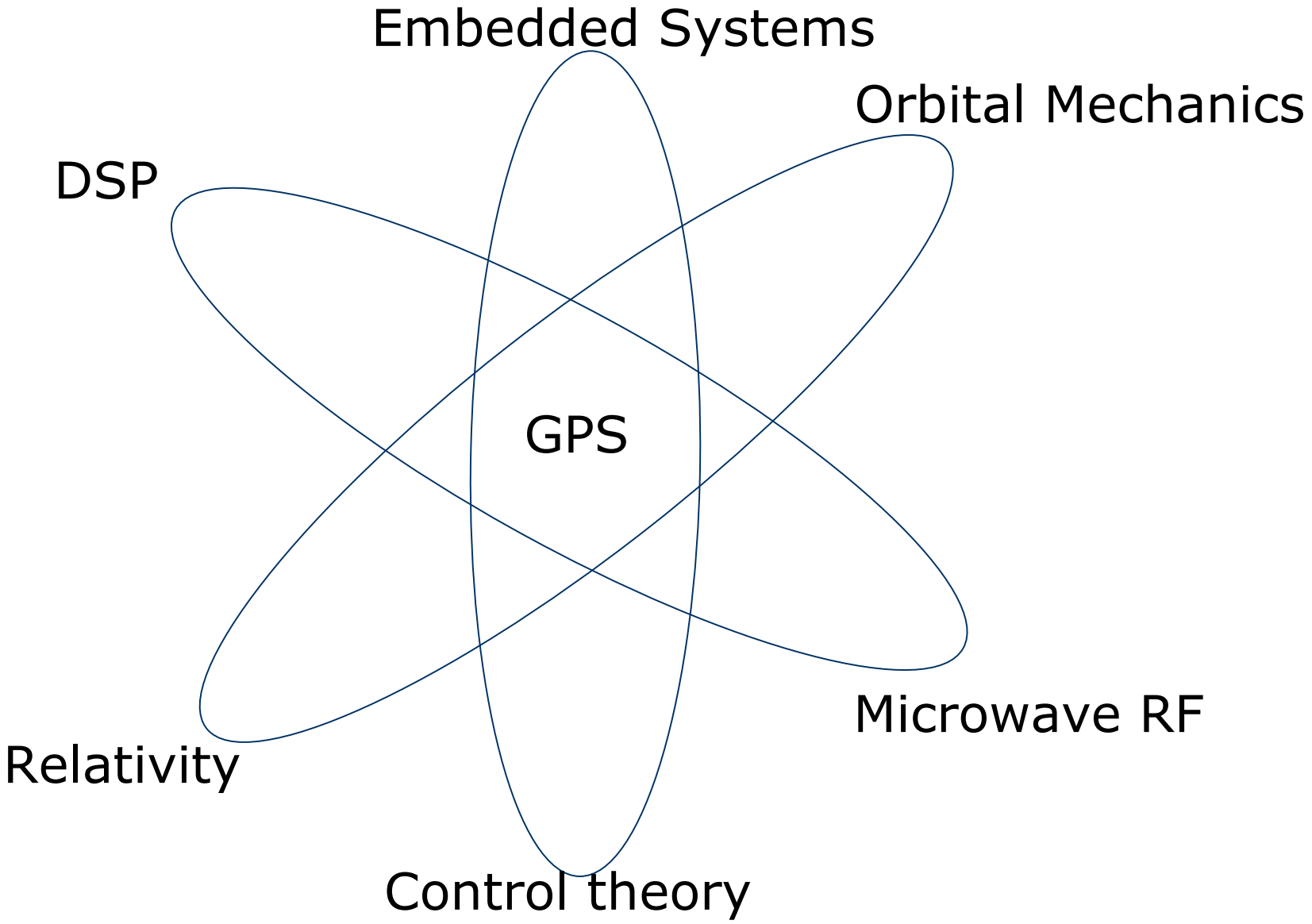
Why Open Source?

- Allows authors to contribute their expertise, while giving them access to the sum of all other author's expertise.

- Creates dynamic, surprisingly full featured software because users:
  - Want the ability to modify the software, and
  - Deeply care about their expertise and the code they generate.

- Extremely successful software development model
  - **Linux** operating system
  - **Firefox** web browser
  - **Eclipse** software development tool

GPS: A different sort of constellation

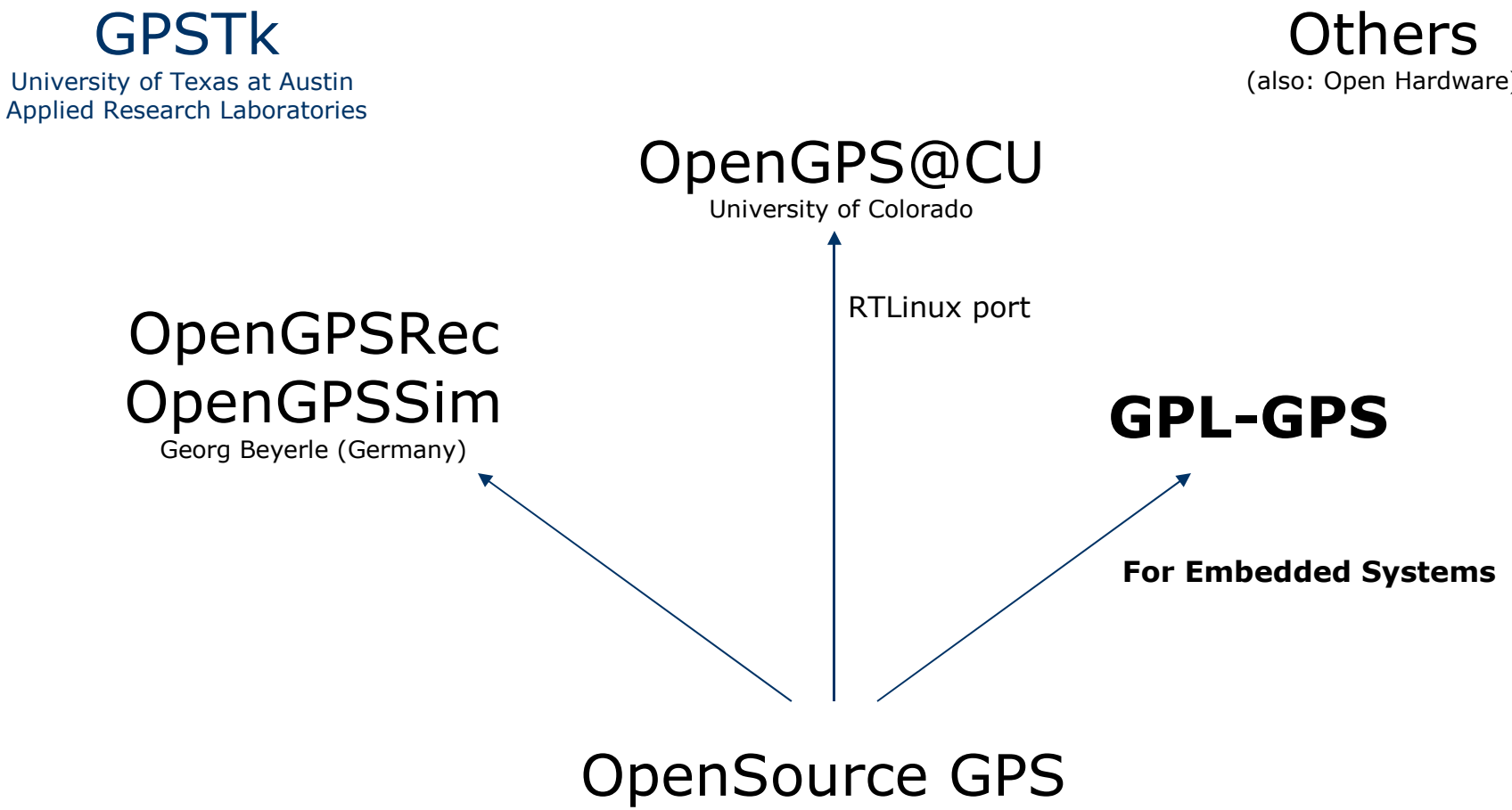


OpenSource GPS

- Dr. Clifford Kelley's **OpenSource GPS**
  - ION GPS 2002 Conference Paper: *OpenSource GPS: Open Source Software for Learning about GPS* by Clifford Kelley, Joel Barnes, Jingrong Cheng

- Free and open source software which controls a Zarlink GP2021 correlator mounted on an ISA or PCI card in a x86 PC.

Open Source GNSS Projects



GPL-GPS Basic Concept

- Take a inexpensive commercial, off-the-shelf (COTS) GPS receiver.
- Erase its original proprietary software.
- Using only free and open source software development tools, program the receiver with our own open source code.

Motivations behind GPL-GPS

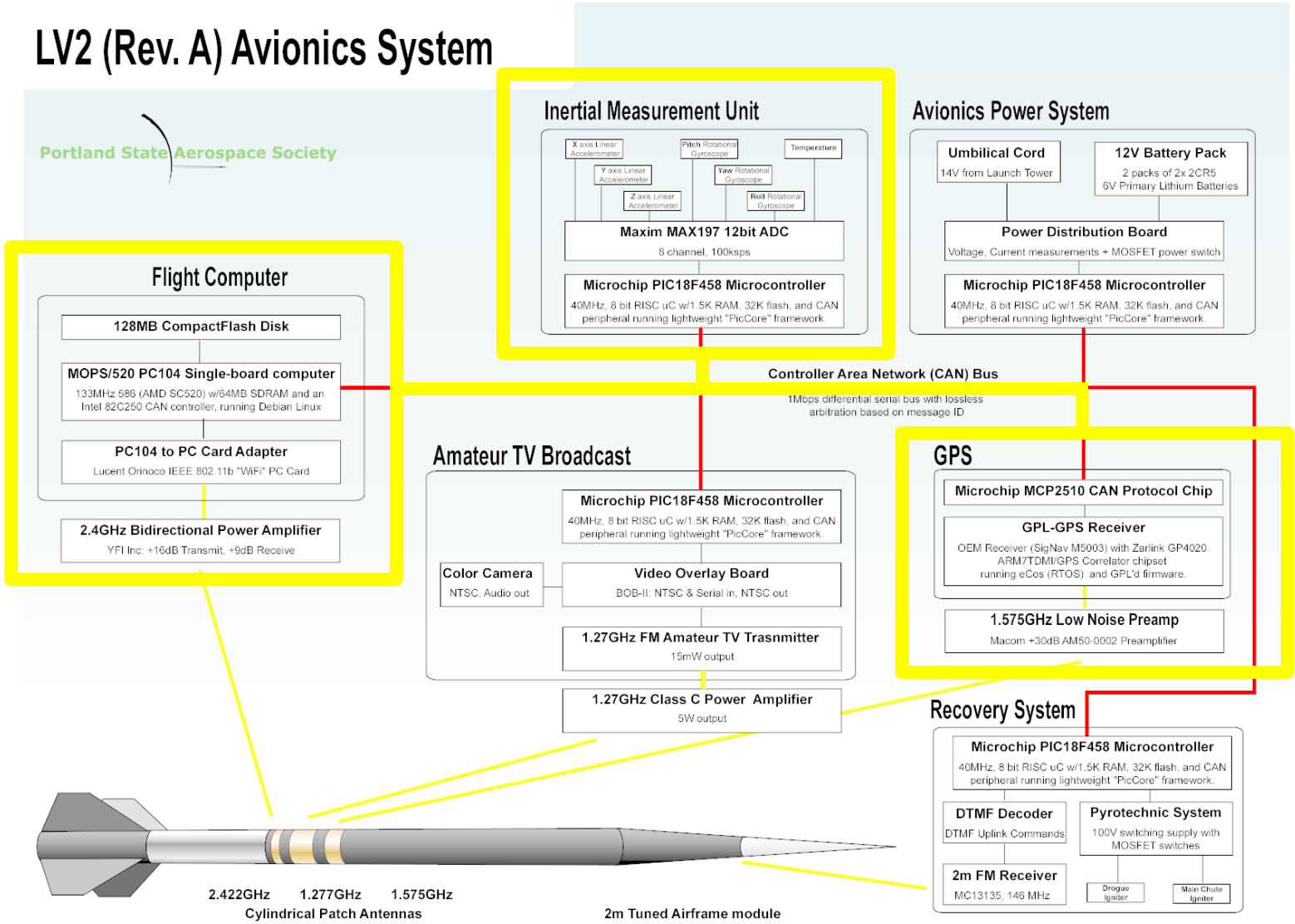
Why Bother?

- COTS receivers generate pseudoranges and satellite navigation data
- COTS receivers have development kits
- What’s special about a a COTS receiver running open source software?

Motivation

- GPS chipset development kits are restrictive:
  - Prohibitively expensive (> US $20,000)
  - **Non-disclosure agreements prevent sharing code**
- Want access to correlators, not just pseudoranges
  - for high dynamics, aiding, research, etc.
- **Reduce barriers to entry:**
  - Extremely Low-cost: uses inexpensive COTS receivers and small carrier boards
  - High participation: Linux, Windows and Mac OS development environments

Original Motivation: Sounding Rockets



Further Motivations

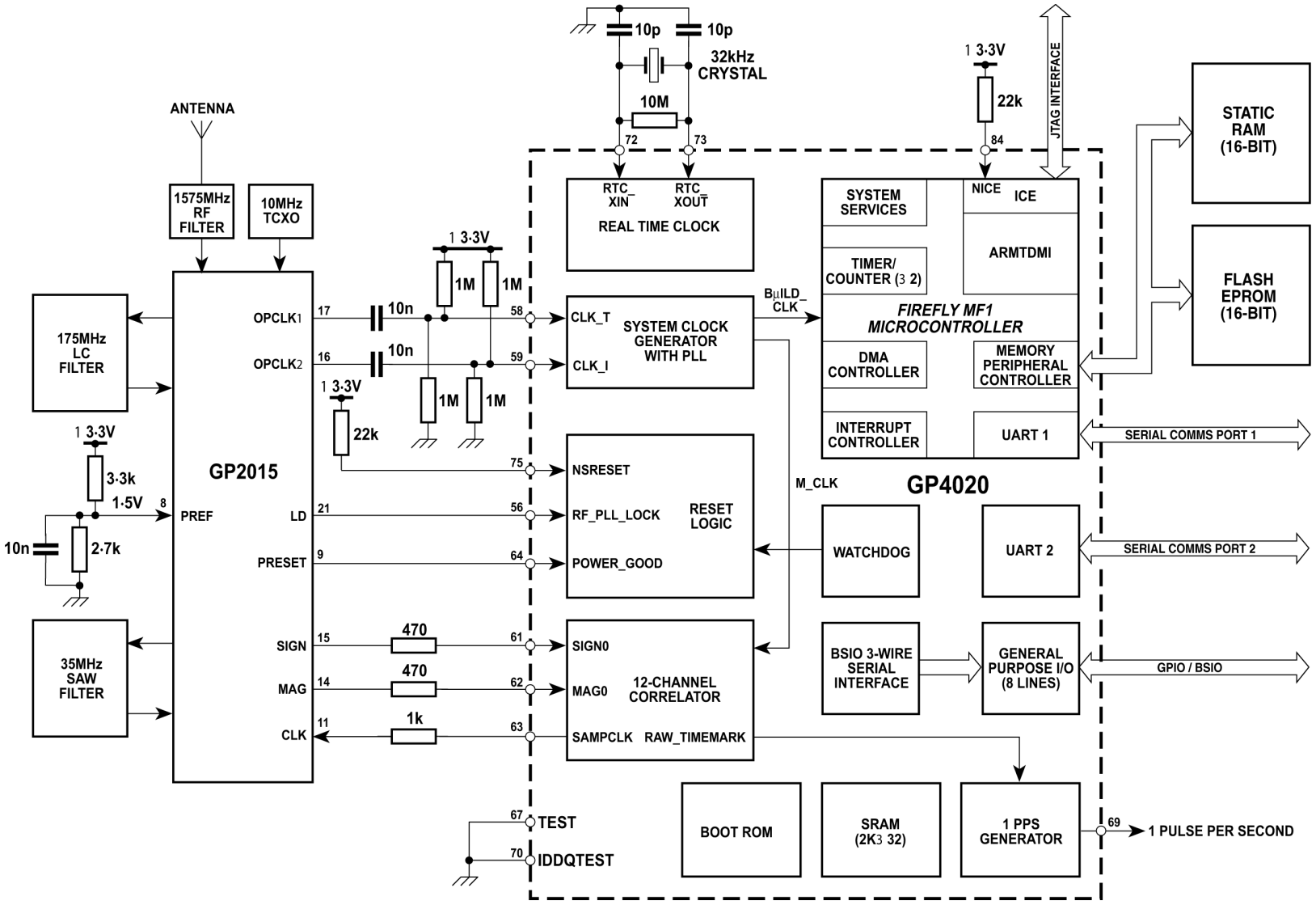
- Lower barrier to entry for:
  - **Nanosatellites**
  - UAVs
  - Pseudolite receiver (dynamic PRN codes)
  - Precision timing
  - Novell applications…
- Creates an open reference system, and a platform for comparing receiver and navigation algorithms in the “real world”.

Choosing a Chipset and Receiver

Choosing a Chipset

- Only *one* GPS chipset manufacturers has open documentation.
- **Why?** Especially on older chipsets?
- Zarlink (AKA Mitel AKA GEC Plessey) Semi.
  - GP2021 (GPS correlator)
  - GP4020 (ARM7TDMI and GPS correlator)
  - Both 12 channel single frequency L1 C/A chipsets

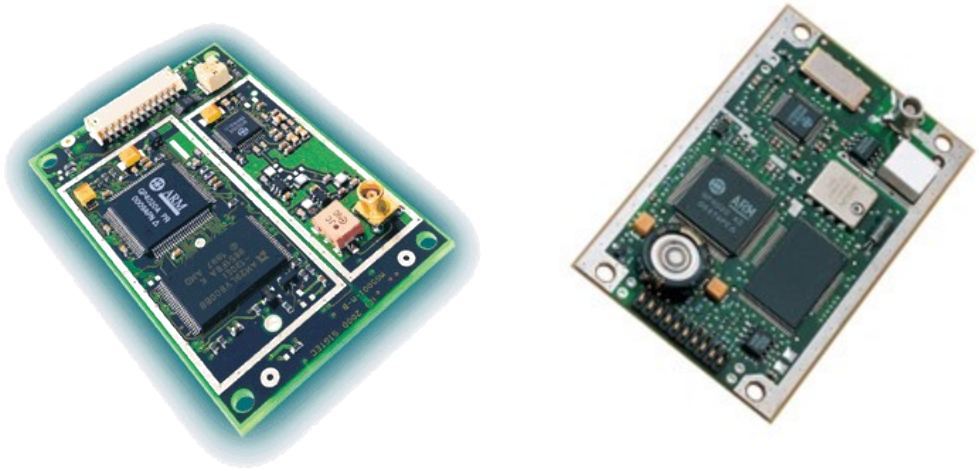
Zarlink GP4020-based Receiver



From: Zarlink GP4020 Baseband Processor Design Manual

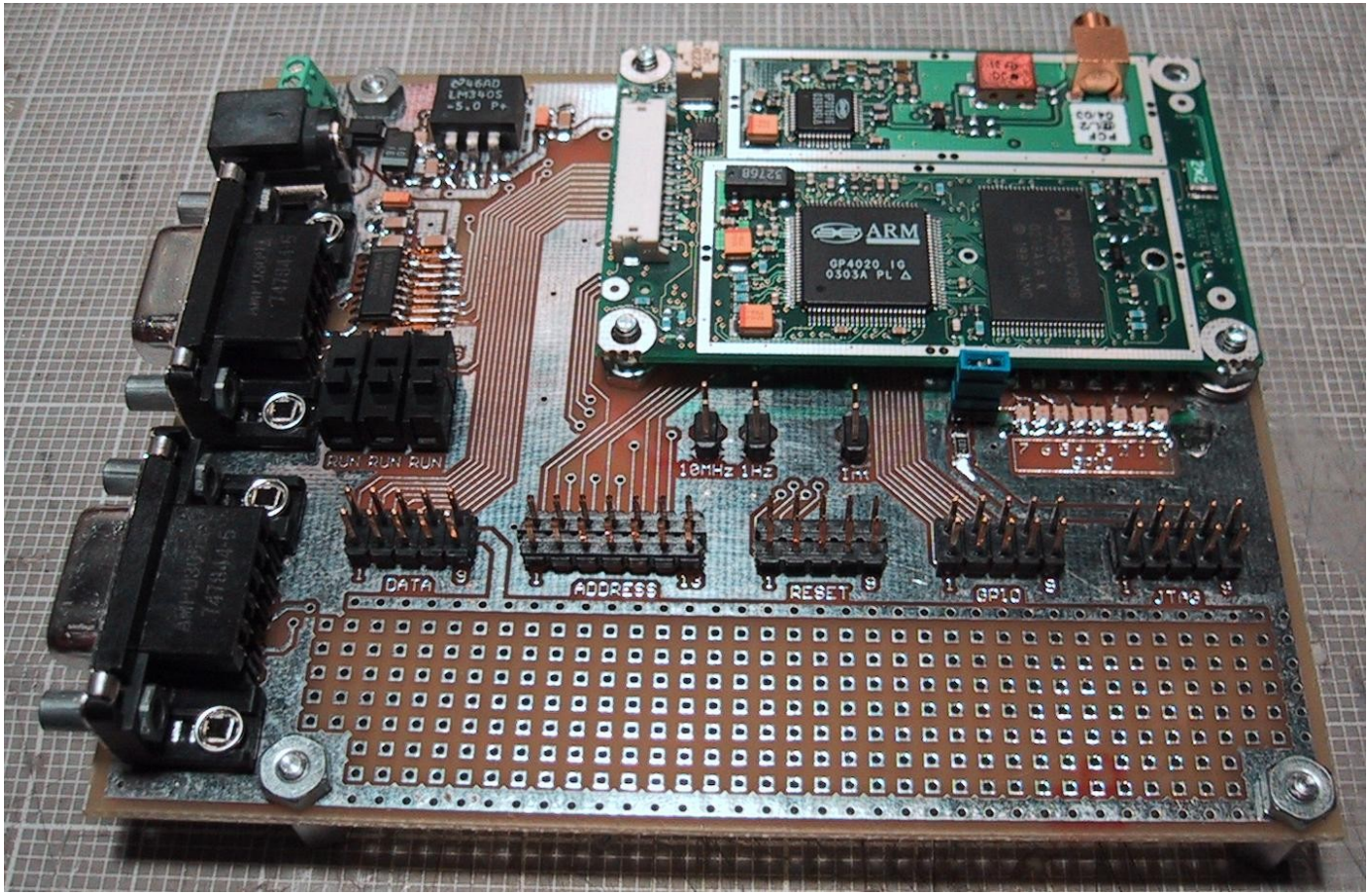
Choosing a GP4020-based Receiver

- Only two COTS GP4020-based receivers:
  - SigNav MG5001 and MG5003
    - DLR “Phoenix” receiver
    - SSTL “SGR-05” receiver
  - NovAtel SuperStar II



GPL-GPS Development Board

- < $50 including parts
- Open hardware: Schematics, board files, parts lists online.



Choosing a RTOS

Real-time operating system (RTOS)

- Why even bother with an RTOS?
- Once ported, has great advantages:
  - Multi-threaded abstraction greatly simplifies software
  - Provides common routines and interfaces:
    - Timers, peripheral drivers, threads, IPC, etc.
    - Insulates the developer from the specific hardware
  - Takes care of “housekeeping” tasks
  - **Makes embedded code development and debugging sane and manageable**

RTOS Comparison (Top 7)

Name	HRT	Small	Free	OSS	x86,PPC,ARM	POSIX	Effort
eCos	Y	Y	Y	Y	Y	Y	4x
µITRON	Y	Y	Y	Y	Y	N	4x
Custom	Y	Y	Y	Y	N	N	10x
µC/OS-II	Y	Y	N	Y	Y	N	2x
Nucleus	Y	Y	N	N	Y	Y	2x
RTX	Y	Y	N	N	Y	N	2x
ucLinux	N	N	Y	Y	Y	Y	1x

HRT = Hard Real Time, Small = Less than 256KB, Free = No fee or licensing conflicts,

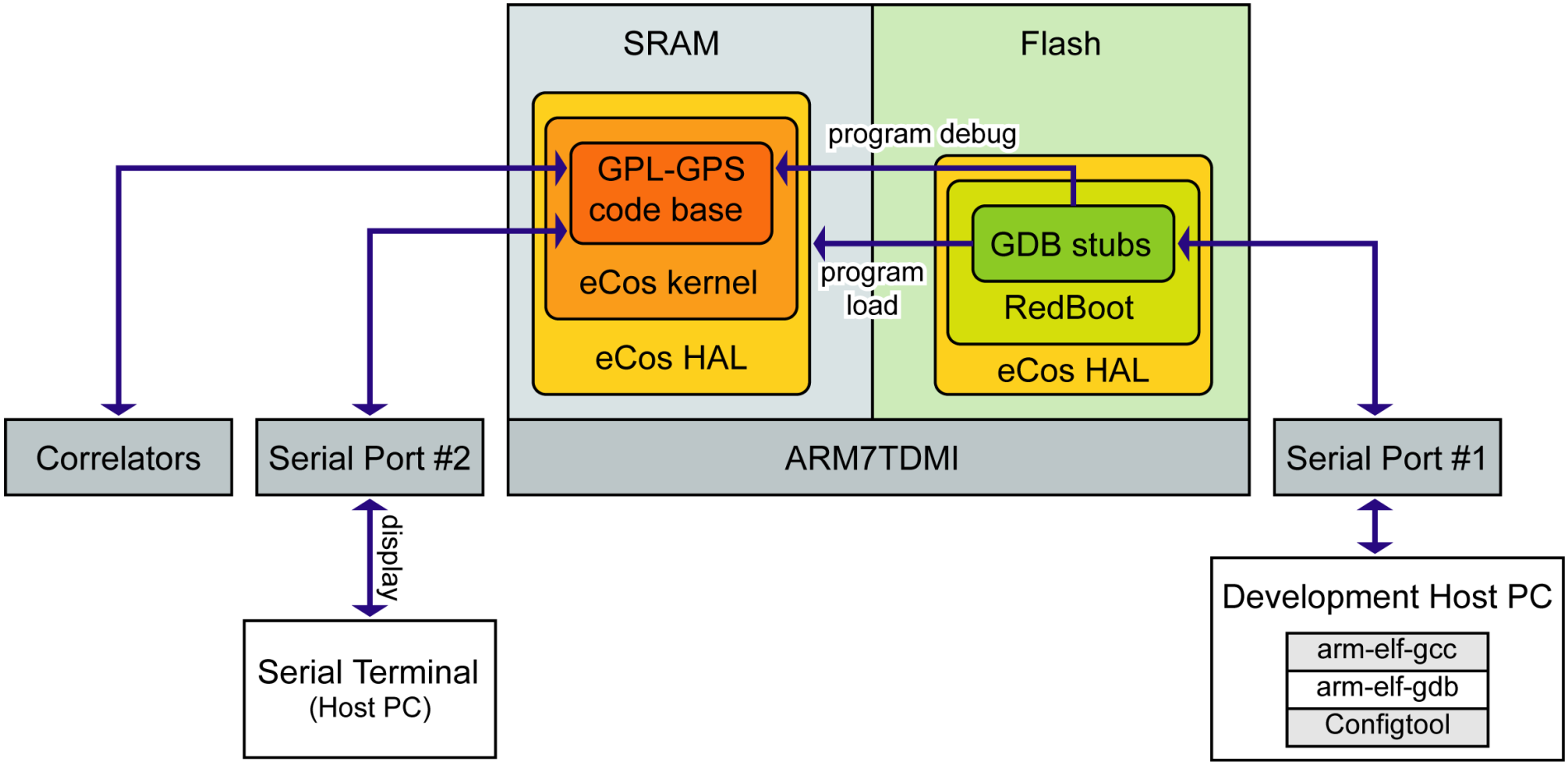
OSS = Open Source Software, x86,PPC,ARM = works on all three targets,

POSIX = POSIX compatibility layer, Effort = effort to port, based on ucLinux port = 1x effort

eCos: embedded Configurable OS

- Uses the open source GNU toolchain
  - gcc, gdb, ld, make, objcopy, etc.
- Works under:
  - Linux (native)
  - Windows (via Cygwin project)
  - Mac OS (requires tool chain compilation)
- Supports almost every 32bit processor, and many 16 bit
- Very light weight (~80 KB), very low interrupt latencies and task switch times.

eCos Infrastructure



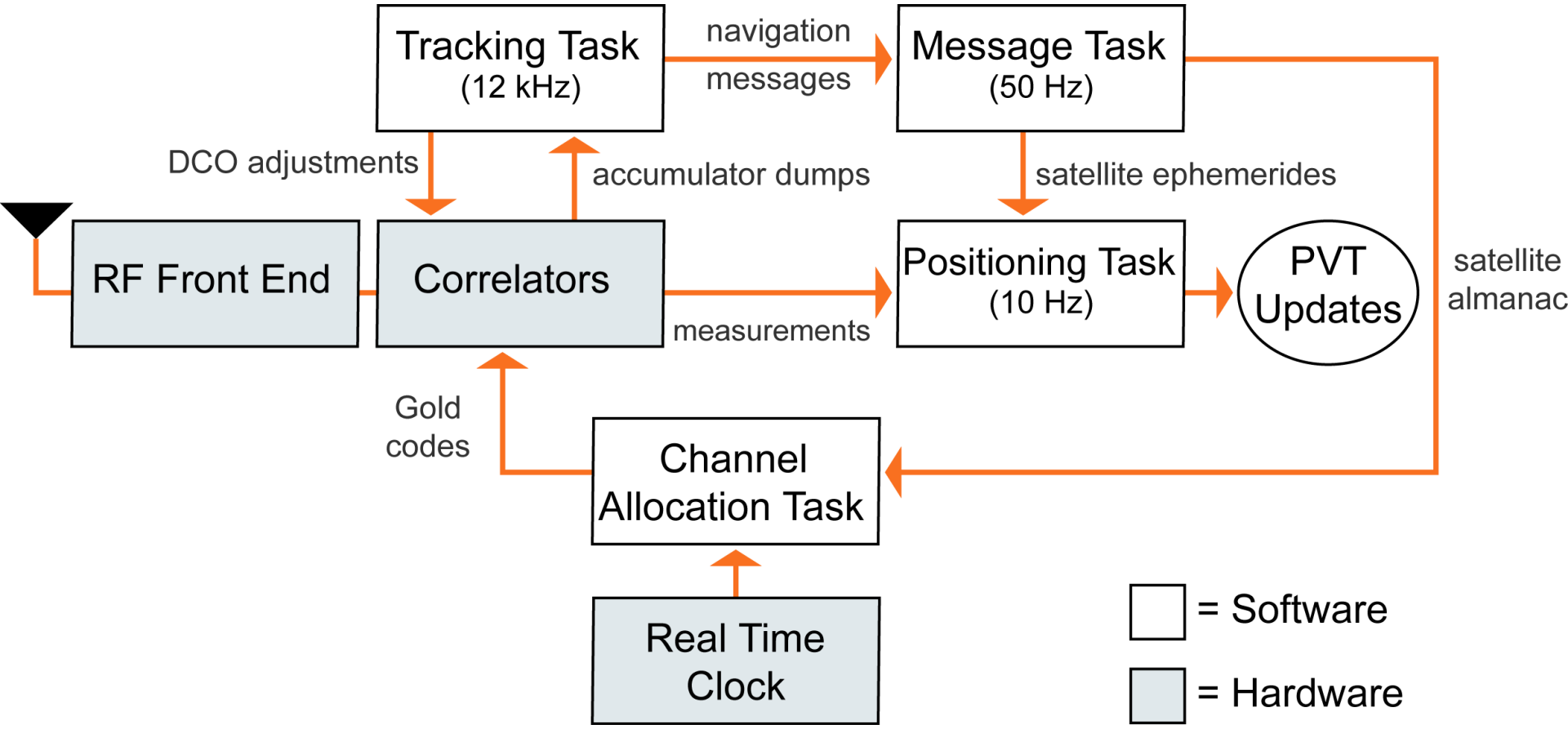
GPL-GPS Development System

- Minimal Hardware
  - SigNav MG5001
  - GPL-GPS Carrier Board
  - Host PC (Linux, Windows or Mac OS X)

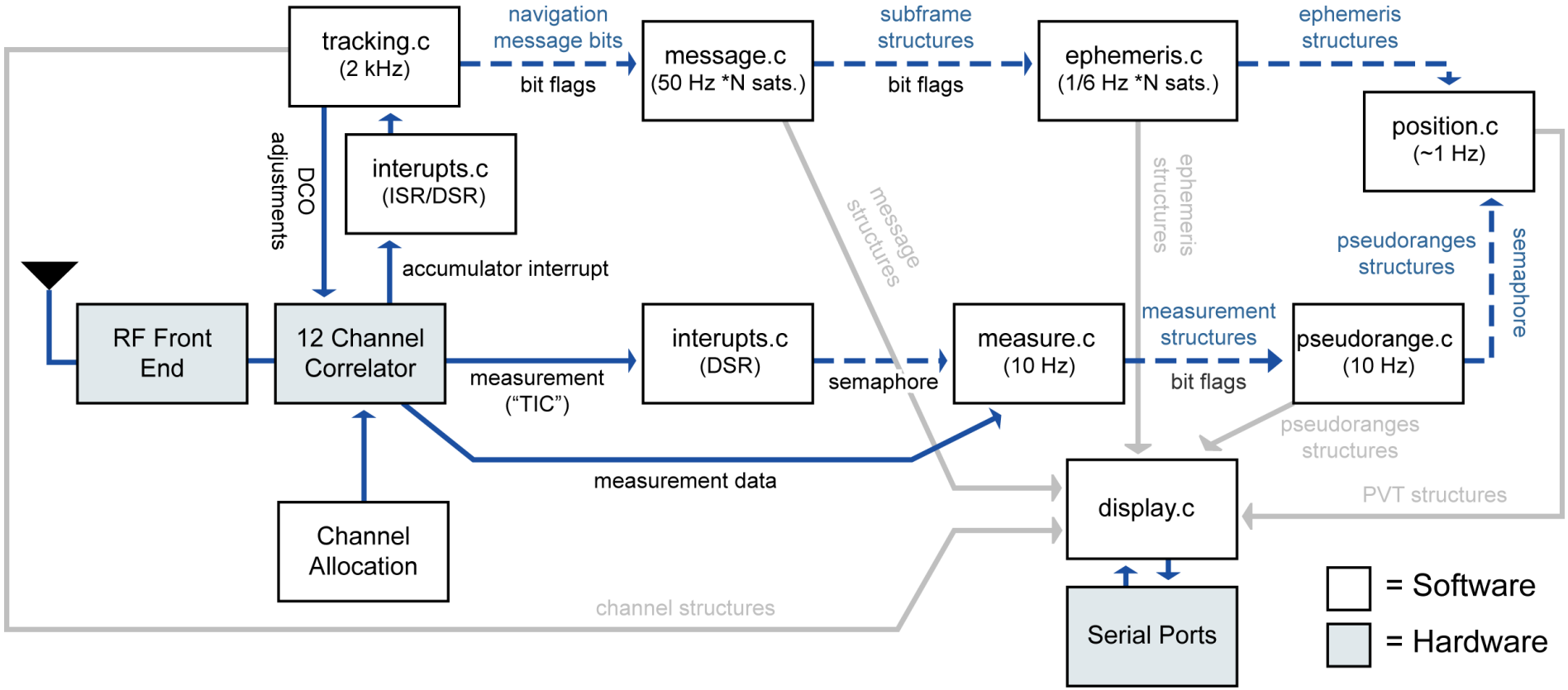
- Free and Open Source Software
  - eCos RTOS
  - GNU tool chain
  - GPL-GPS Software

GPL-GPS Software

Generic GPS Receiver Tasks

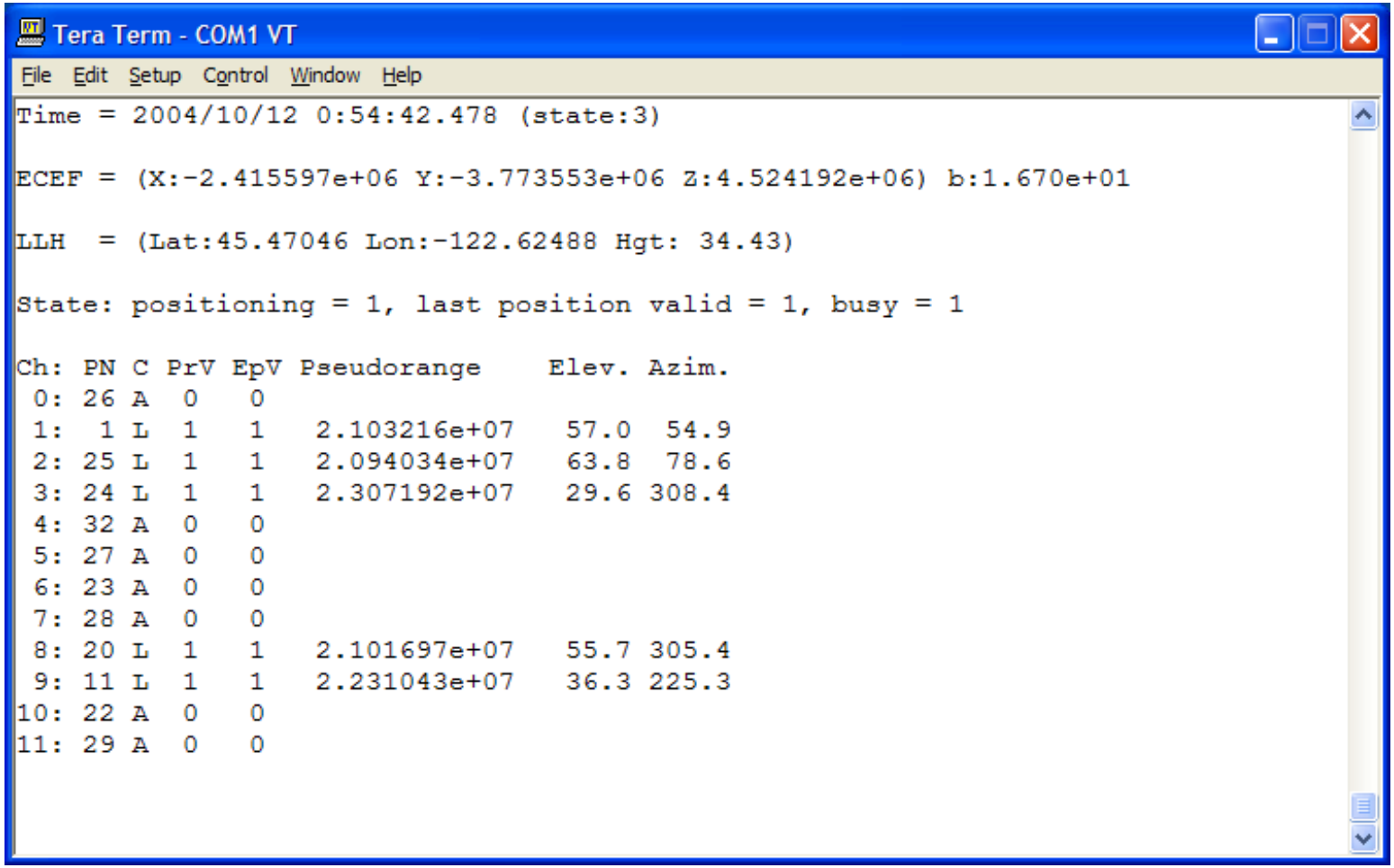


GPL-GPS Tasks



Status and Initial Results

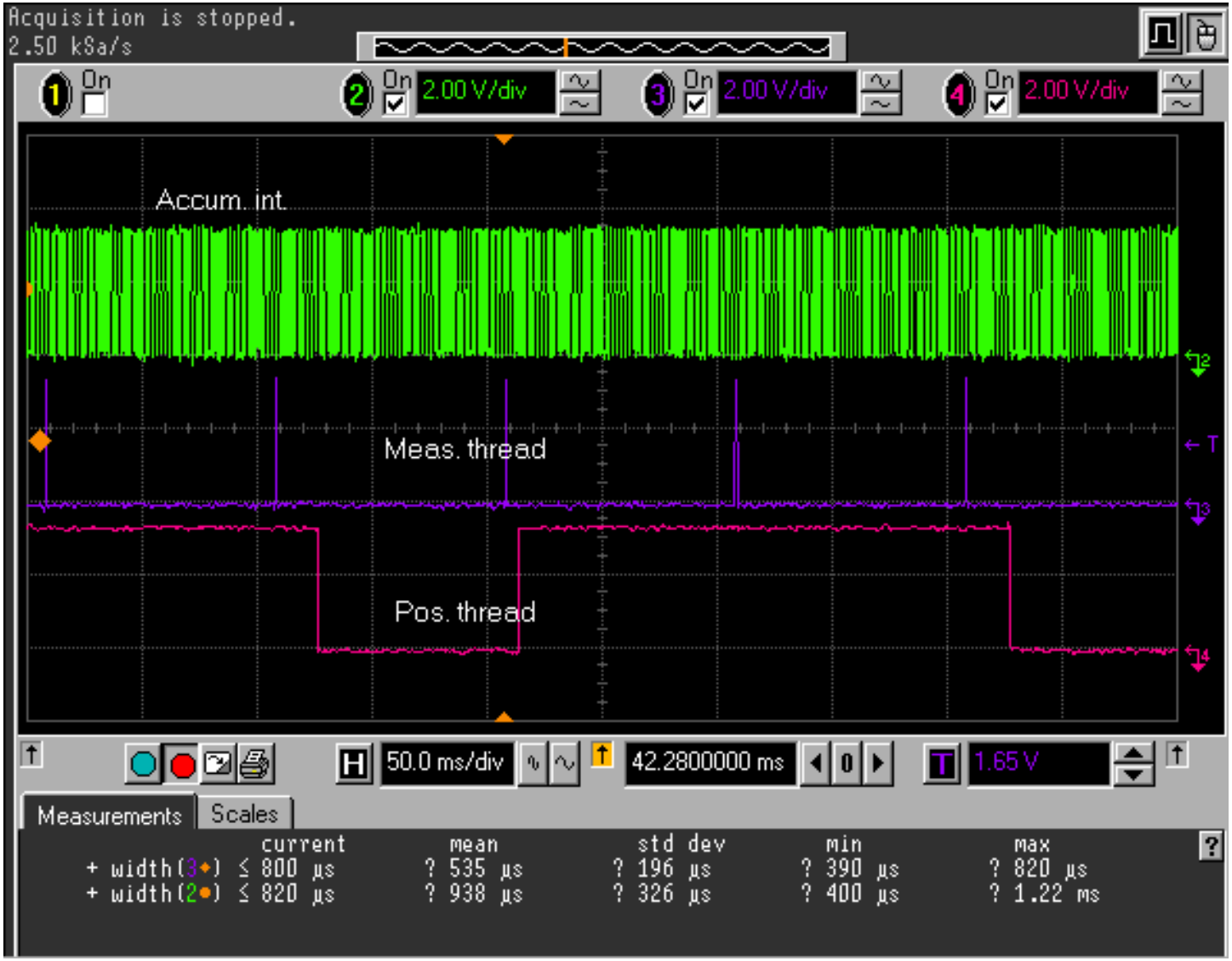
First Fix May 2nd, 2005



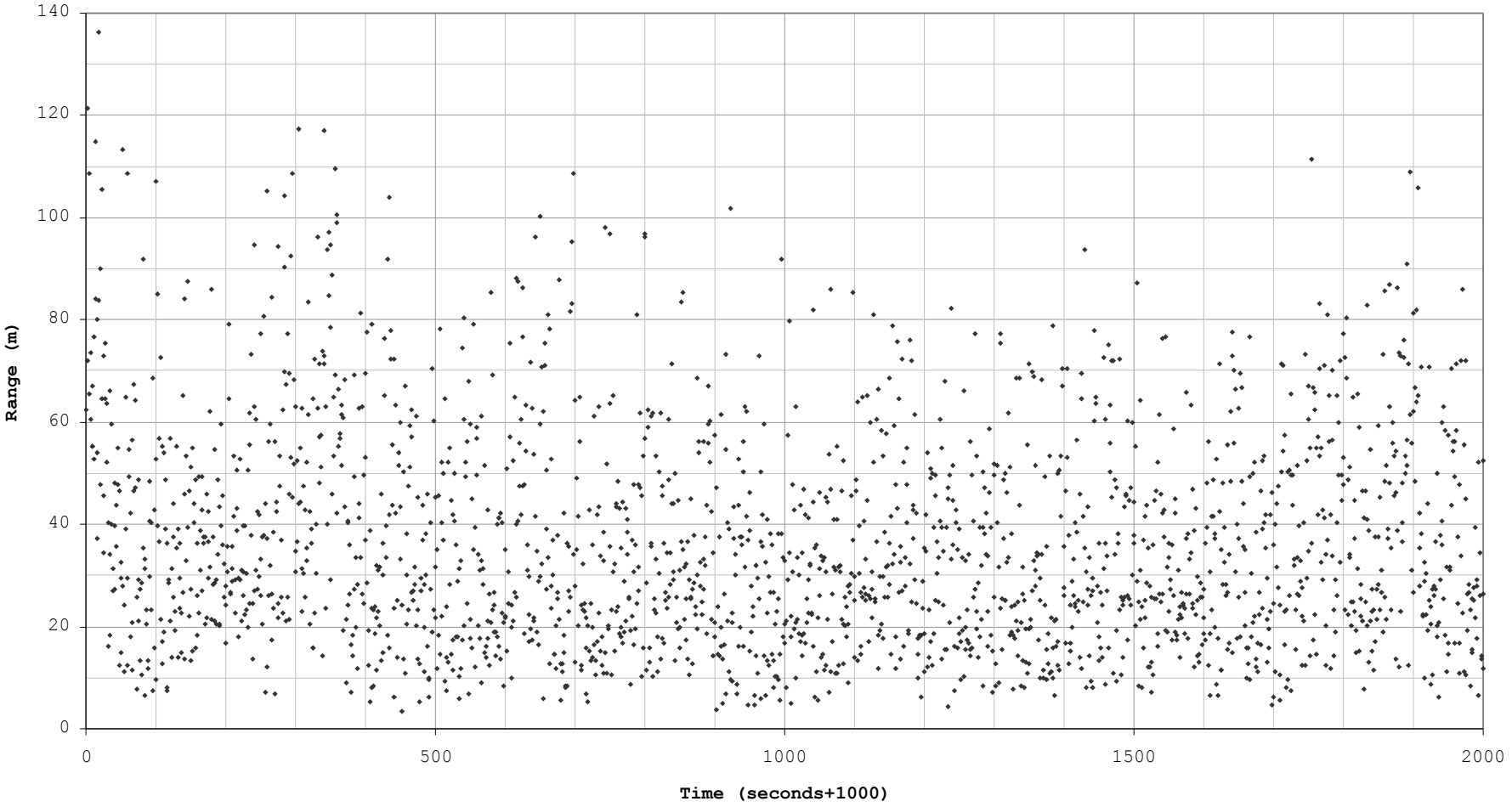
GPL-GPS Current Status

- It works!
- But:
  - Lacks atmospheric correction
  - Lacks carrier phase
  - Poor tracking loops (2nd order PLL)
  - Poor acquisition and search on loss-of-signal
  - Almanac not stored in non-volatile memory
  - eCos infrastructure needs more tuning

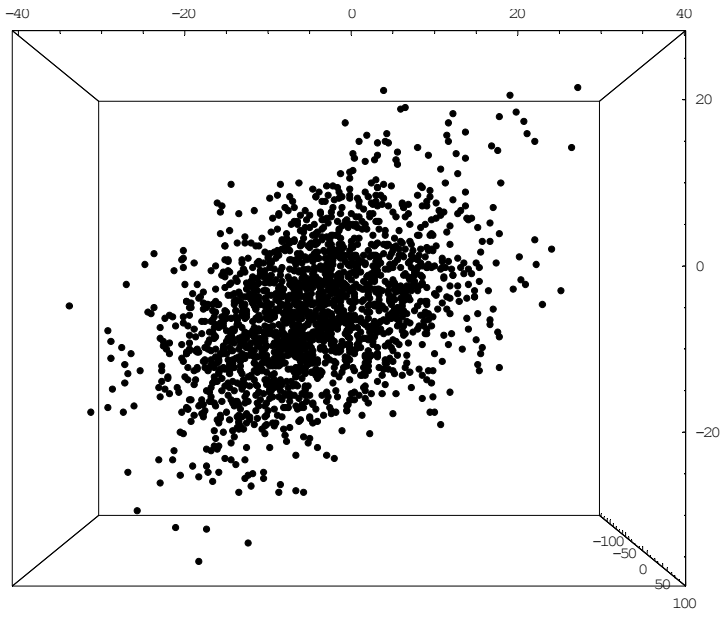
Processor Performance

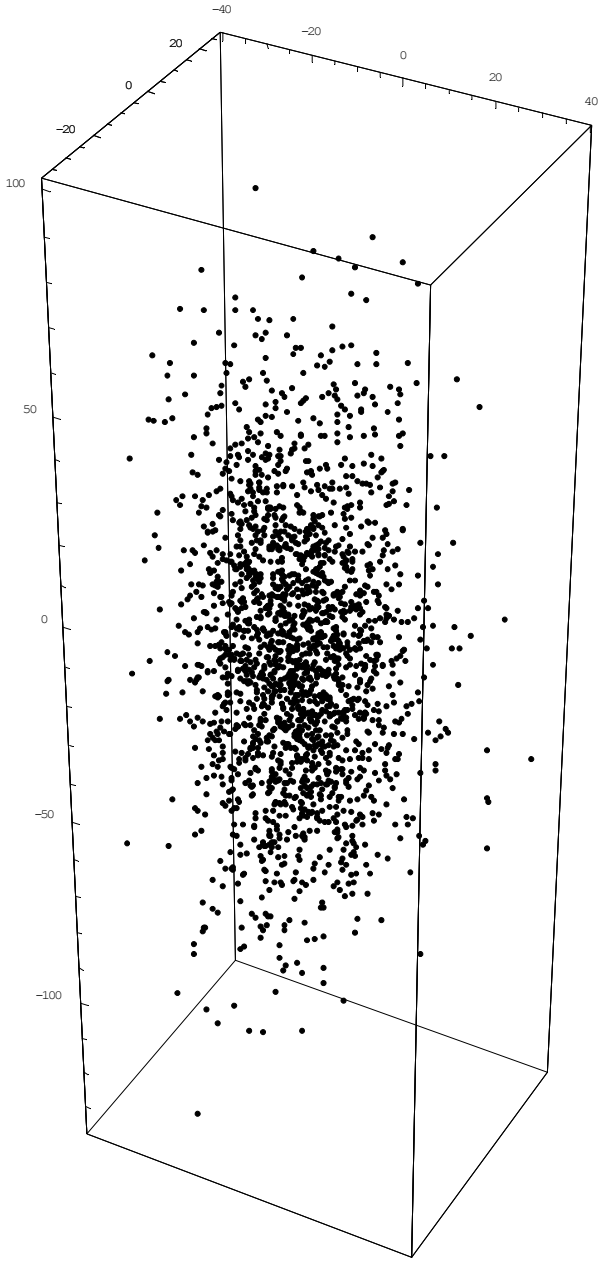


Static Performance (2000 pts., 36.4m)

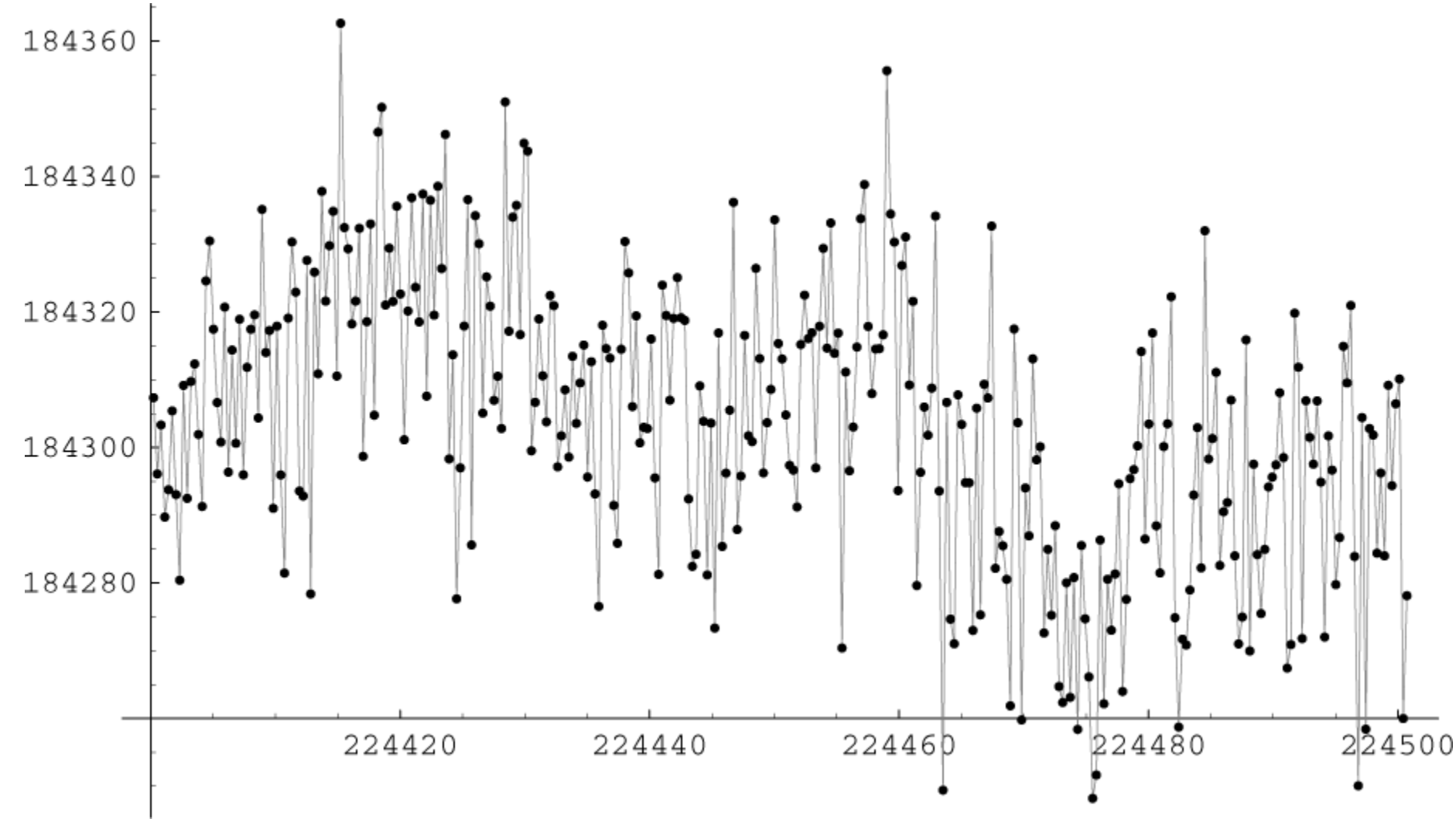


Static Performance: 3D

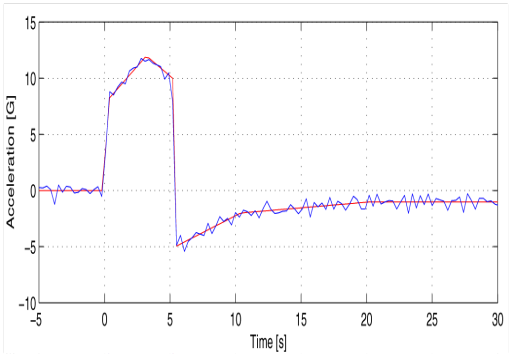


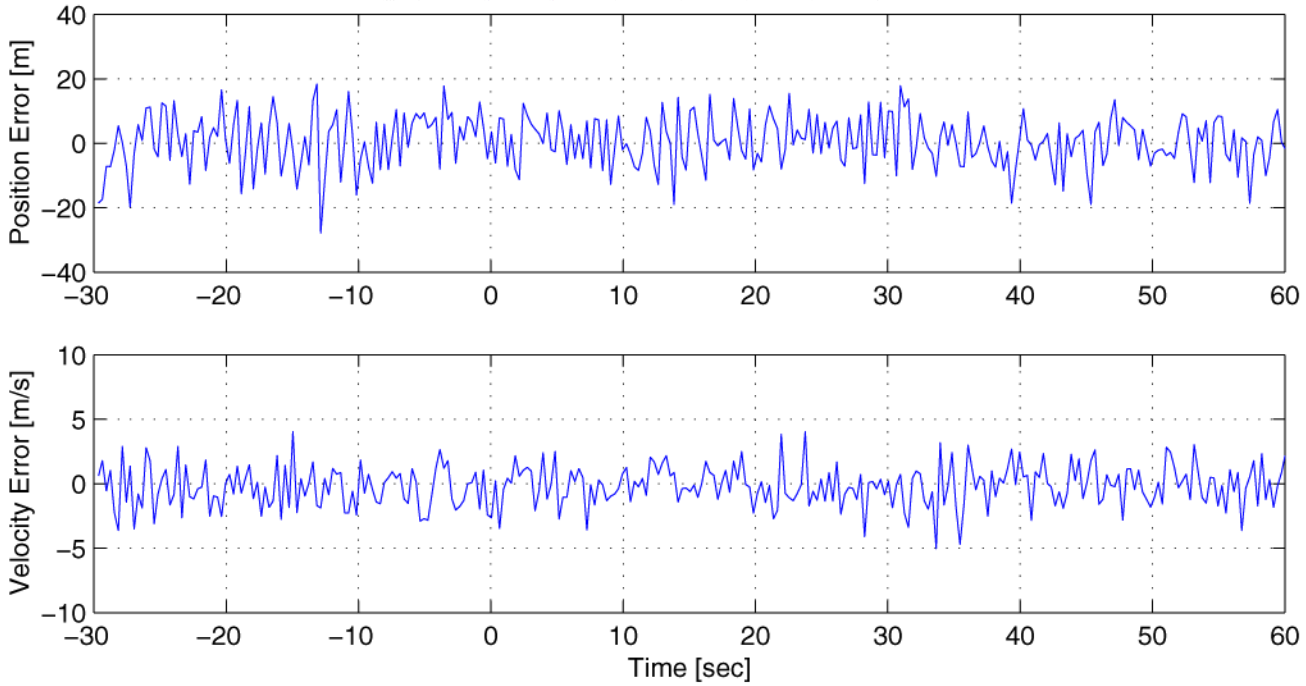


Static Performance: Time Series

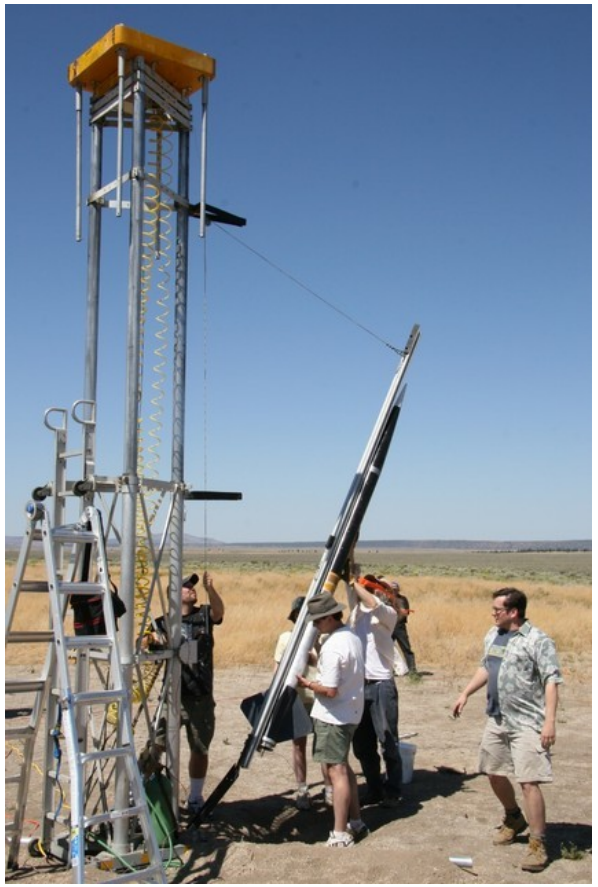


Simulated High Dynamics Performance

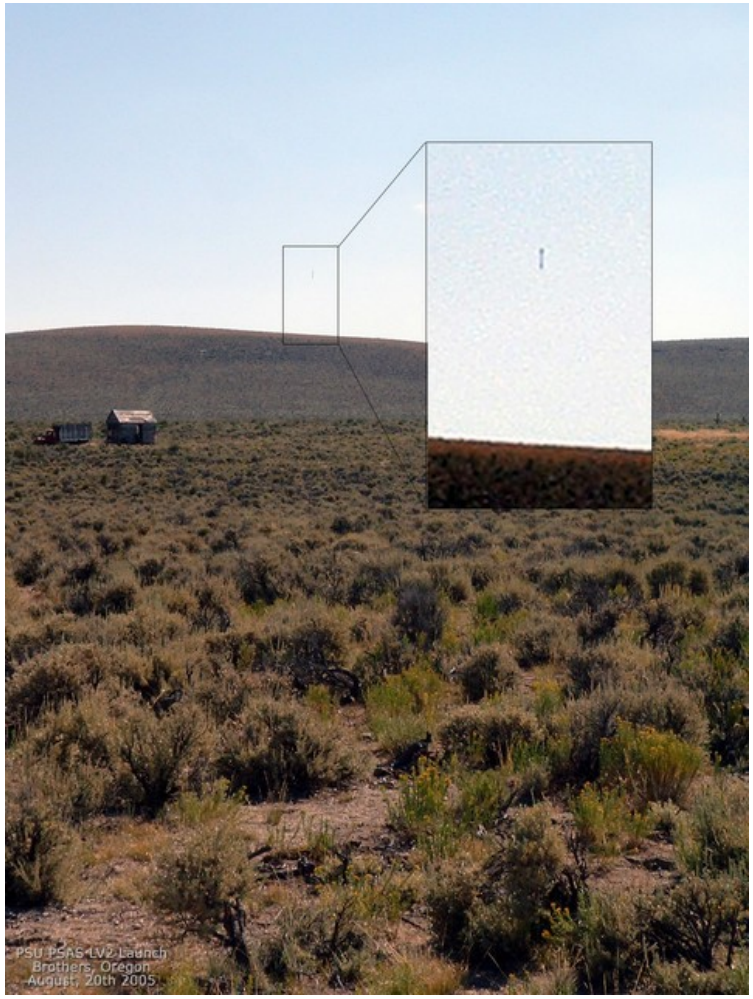




High Dynamics Test Platform



HD Test Flight… slightly delayed



Future Work

- Infrastructure work
  - Fix bugs. Probably lots of them
  - Move interrupts & tracking to internal fast SRAM
  - Implement GP4020 PPS hardware
  - Network API for external PC interface
- Reimplement current OSGPS features
  - Atmospheric correction
  - Carrier phase
  - Almanac load/save and processing (via Flash)
  - Faster/better cold acquisition

Conclusion

Conclusion

- Created an open source software development system for commercial off-the-shelf GPS receivers
- GPL-GPS promises to dramatically lower the high barrier to entry for GPS development

The GPL-GPS Project
http://gps.psas.pdx.edu/