A Modular GPS Remote Sensing Software Receiver for Small Platforms

Marcus Junered¹, Stephan Esterhuizen², Dennis Akos², Penina Axelrad²

¹Luleå University of Technology, Luleå, Sweden
²University of Colorado Boulder, Colorado, USA

Abstract

This paper focuses on the processing of experimental data collected with a small modular GPS bistatic radar software receiver on a balloon flight. The receiver is designed for remote sensing purposes and the design process will be covered in some detail. GPS bistatic radar focuses on the surface-reflected signal, which can be used to determine properties of the reflection surface, including roughness, ocean wave height and also ranging to the surface. In the past, a mixture of both ASIC-based and software receivers have been used, but these require desktop or laptop computers to operate them. Previous receiver configurations have been prohibitively large and could not be mounted on smaller platforms. This design features two analog front-ends with a common clock, a FPGA and USB bridge to move the digital samples to a Linux-based Single Board Computer. The system features a modular design and allows for easy integration with other analog front-ends. Airborne data collected with this instrument near Boulder, Colorado during a balloon flight will be presented. The data from the balloon flight has been processed to extract the height above ground using GPS bistatic radar as an altimeter and the GPS position as a reference. GTOPO30 Digital Elevation Models have been used to estimate the surface elevation used for height estimates in the altimeter. This project opens up new opportunities to perform remote sensing with cheaper and smaller platforms.

1 Introduction

GPS signals has been used for opportunistic bistatic radar for some time now, mainly for remote sensing of the earth’s surface. Typical applications range from altimetry [6] to determining surface roughness [4], soil moisture and ocean wave height [1], all based on processing of the reflected signal. Earlier studies have shown that GPS reflections can be detected even from space [2]. The principle for a GPS bistatic radar is simple: by using two channels: one tracking the direct signal from the satellite; the other, slaved to the direct channel looking for the surface-reflected, delayed, signal. The direct channel is connected to a skyward oriented antenna, and the reflected channel to an antenna oriented toward the earth. Traditional hardware for GPS bistatic radar [6, 4] has consisted of custom receivers with one common factor, they all required at least a laptop computer to operate. One of the drawbacks of having to use a laptop or desktop/rackmount computer is that the receiver becomes heavy and bulky and needs to be flown on a plane of significant size, eg. a manned plane. This design on the other hand focuses on a miniaturized receiver put together with mostly commercial off the shelf components to reduce the cost [7, 8]. The design is built around two GPS L1 front-ends with a common clock, connected to a USB bridge for high-speed data transfer. Instead of using a laptop computer a single board computer (SBC) in the “nano ITX” form factor was used. The computer is running an embedded version of Linux and most of the software used comes from the GNU Radio project [9]. The first section of this paper covers the design construction and validation of a small modular GPS bistatic radar software receiver on both the hardware and software side. Section 3 covers the basics of GPS bistatic radar and gives an introduction to radar altimetry with GPS reflections. The next section describes the details of a balloon flight used as a test for the receiver, and the post-processing of the experimental flight data. It also contains a comparison between the GPS bistatic radar altimeter and the vertical component of GPS position solutions from a Commercial Off The Shelf (COTS) software receiver. The last section, section 5, contains a summary of the design and discusses the altimeter’s performance.
2 Receiver Design

The receiver designed consists of three parts, the front-ends which down-converts the GPS signal to an IF and then samples it. These samples are then buffered by a USB data bridge for retrieval by a host computer and stored to disk (see figure 1). All the components shown in figure 1 are commercially available except for the interface board, which had to be custom designed to provide a backplane for connecting the components together. See [8, 7] for more details on the design.

2.1 GPS Front-ends

The front-ends (figure 2) used were SiGe Semiconductor SE4410L v.1.1 [17] evaluation boards. The front-end downconverts the GPS signal at 1575.42MHz to an IF of 4.1304MHz and then samples it at 16.3676MHz using 2-bits. Both front-ends operate off the same clock which is fed from one board into the other through the external reference input. A more detailed illustration of how the boards are connected can be found in figure 3.

The data bits pass through both the FPGA and the FX2 USB chip. This way the receiver has the ability to operate either with or without the FPGA installed.

2.2 FPGA

The front-end IF samples optionally passes through an Altera FPGA [19] for further processing. The FPGA is currently not in use and IF samples are bypassed (as seen in figure 1). Since a bi-directional bus exists between the FX2 USB chip and the FPGA, the FPGA can conceivably be used in the future to implement real-time signal processing algorithms. However, the receiver is fully functional even without the FPGA board.

2.3 USB Data Bridge

The USB data bridge [10, 9] is used to move high-speed IF samples from the front-end (or FPGA) to the host computer. This data bridge has been benchmarked and can reliably move data at up to 39MB/s on a modern (1GHz+) computer. The Cypress Semiconductor CY7C68013 FX2 USB chip takes 8 or 16-bit data at the rising edge of a clock (CLK OUT) and stores this data in a FIFO to be packetized by the on-chip USB handler.

2.3.1 Firmware

A portion of the firmware that runs on the FX2 chip has been written by GNU Radio [9] developers for the SDCC compiler [11] and the fx2-programmer software [12]. This firmware was heavily modified to accommodate for the data transfer method between the front-ends and the FX2 chip. For a better understanding of the FX2 operation, see figure 4. The FX2 chip consists of four parts, the General Programmable Interface (GPIF), buffers, an 8051 microcontroller, and a USB engine.
2.3.2 General Programmable Interface (GPIF)

The GPIF is a general purpose state machine that can be programmed. In the case of the digital stream from the analog front-end chips, the interface is simple: On the rising edge of the front-end CLK OUT signal, 4 data bits become available (2-bits per front-end). Once the GPIF is started by the host code, it continually puts data into the buffers until either the host code is stopped or the buffer fills up, this means that the 8051 does not participate in the data transfer in any way. Data is automatically put in the buffers and when a buffer is full (has reached the maximum payload for a USB packet) it is committed to the USB engine for packetization.

2.3.3 Buffers

The firmware sets up the GPIF to connect to end point 6 (EP6). EP6 in the FX2 is configured as quad-buffered with each buffer being 512 bytes. The buffers can be filled and emptied in parallel, allowing for a smooth transfer of data between the GPIF and the host computer.

2.3.4 8051 Microcontroller

The 8051 microcontroller is responsible for initializing the FX2 chip and monitoring GPIF status, the host-code periodically polls the GPIF status register to ensure that the GPIF is not idle (which would indicate a buffer overrun).

2.3.5 USB Engine

The task of the USB engine is simple: As soon as a 512-byte buffer is filled, prepare the contents of the buffer for a USB transfer and then wait for the host software to request data. USB has a strict master/slave protocol and the slave (the FX2 chip) can not send data to the host unless the host requests data first.

2.3.6 Host-side Software

The C++ host-side software works on both Windows and Linux, however due to performance requirements an embedded version of Gentoo Linux [15] is installed on the SBC. It uses basic function calls to libusb [13], a library that allows access to USB devices. Bulk transfer mode was selected because the general USB community indicates this to give the highest reliable USB throughput [14].

2.4 Single Board Computer

The single board computer runs the host-side USB code. The host computer features a low-power VIA Luke Core-Fusion Processor [16] in the Nano-ITX form factor (12cm x 12cm). This is an x86 compatible processor that runs at 1.0GHz and supports most modern peripherals, including Serial-ATA (SATA) and USB 2.0. The SBC has 4 USB connectors and boots a small version of Gentoo Linux [15] from a 128MB USB disk. The SBC runs off a normal ATX power supply.

2.5 Storage

With a sampling frequency of approximately 16MHz and with 4-bits of data (in 2-bit mode) for two channels, the data rate is about 8MB/s. Due to possible vibrations of small platforms (like UAVs), a mechanical drive might not suffice for storage of IF samples. Two other solutions were investigated, a 2.5 solid state disk (SSD) laptop drive, and USB memory sticks.

2.5.1 2.5” Laptop SATA SSD

It is advantageous to use a 2.5” SSD because it has the same form factor and data interface (SATA) as other mechanical laptop hard drives. Thus, cheaper mechanical hard drives can be used when the platform is not operating within a vibration-rich environment. A good SSD can sustain about 30 MB/s transfer rates.

2.5.2 USB Memory Disk

This is a more cost effective solution because USB memory disks are popular mainstream devices. These pose a difficulty to use as a storage device because the USB bus must be shared between the receiver and the USB disk. Two 4GB USB disks can be combined using software RAID0 to form a 8GB partition which can sustain about 11 MB/s.

2.6 Interface Board

All the aforementioned modules must be connected together. This is where the interface board [7] comes in. It is a custom
designed 4-layer printed circuit board (PCB). The two analog front-ends plug into the back-side of the board, the digital components (FX2 USB and FPGA) plug into the front-side of the board. The two inner planes are power and ground planes and thus provide some shielding between the analog and digital circuitry.

3 GPS Reflections and Radar Altimetry

3.1 Specular Point and Geometry

For a detailed description of GPS bistatic radar see [5]. GPS bistatic radar build on the fact that the signal from a satellite will reflect off the earth’s surface in such a way that it can be detected by a down-looking antenna on an airplane or other elevated platform. In order for the receiver to be able to detect the reflection it needs information about when to expect the reflection, this information can be calculated if the direct signal is being tracked and the satellite’s elevation is known. The reflection can thus be detected in an open-loop fashion by slaving correlators to the prompt correlator and delaying them according to the expected additional signal path the reflected signal has to travel. The specular point is defined as the point on the earth’s surface where the reflection occur, or more specifically the center of the reflection area. Figure 6 illustrates the geometry associated with the specular point.

Figure 6: Specular point and reflection geometry

The specular point can be calculated if the user position, satellite elevation and the path delay is known. A byproduct of the specular point calculation is user height above the specular point.

3.2 GPS Bistatic Radar

With knowledge of the path delay between the direct and the reflected signal a bistatic radar can be created. A traditional monostatic radar has both the transmitter and the receiver in the same location, while a bistatic radar has the transmitter and receiver in different locations. For a GPS bistatic radar the transmitter is the satellite and the receiver is on the vehicle/platform. Advanced GPS Bistatic applications might combine data from several satellites in such a way that it operates more like a multistatic receiver, or a radar with several transmitters and one or more receivers.

3.3 GTOPO30

GTOPO30 is a digital elevation model developed by the U.S. Geological Survey’s Center for Earth Resources Observation and Science (EROS). The horizontal resolution is 30 arc seconds or about 1 km. The model is split up in tiles with a size of 50x40 (lat x lon) degree segments covering the Earth surface. The tile used for this paper was the W140N90.

4 Balloon Flight

A high altitude balloon flight [20] was utilized as a practical test of developed hardware. Student payloads were connected to a cord that was attached to the balloon as shown in figure 8. The balloon would ascend to an altitude of approximately 28km MSL where it would burst and free-fall until there is enough atmosphere for the parachute to deploy reliably. The balloon was tracked via a beacon that transmits GPS position down to the tracking station. The bistatic instrument was mounted approximately 3m above the beacon on the cord.
The balloon flight took approximately 3 hours from launch to landing. The flight path can be seen in figure 9. The balloon traveled a total of 97 km from the launch site. Recovery was successful and no equipment was damaged.

4.1 Radio Frequency Interference (RFI)

Radio Frequency Interference or RFI is any undesirable signal power in the spectrum of interest. RFI will degrade the receiver’s performance. Continuous Wave (CW) interference might show up very clearly in a Power Spectral Density (PSD) plot, however because of the narrow bandwidth the total power is not very high. An example of this can be seen in figure 11, however the GPS receiver did not appear to have degraded performance under these conditions. A clean L1 spectrum (2.2 MHz spectral BW in the front-end), free of any interference can be found in figure 10.

Before any baseband signal processing of the balloon data was done, the spectrum of the IF samples collected during the flight test was analyzed for any RFI. If the antennas are mounted close to the computer some CW spikes are detected as shown in figure 11 and 12 below.

Figure 8: Balloon payload configuration

Figure 9: Balloon flight path

Figure 10: Spectrum free of interference (direct channel)

Figure 11: RFI in PSD for direct channel (Channel A) during balloon flight
Tests with antennas placed further away from computers and other noise sources show no signs of interference. This means that there is no self-generated interference coming from the receiver itself.

### 4.2 Altitude Profile

Telemetry obtained from the beacon on the balloon was used to generate an altitude profile (as seen in figure 13). Due to limited solid state storage (2x4GB USB disks), only 16 minutes of GPS samples could be recorded. A solid state drive could have been used but due to the high cost of such devices (more than the rest of the receiver) two cheap USB disks were used instead. The data collection was broken up in four 4-minute sections as indicted in red on the altitude profile. Reflected data presented in this section were collected from around 15 km (circled in figure 13).

To extract altitude from bistatic measurements the path delays are needed. Figure 15 below shows how the reflected peak moves from having a delay of 69.2 chips to a delay of 72.9 chips during 160 seconds. This is the same time span as the one used for the altitude plots in section 4.3. Since 1 chip is approximately 300 meters in distance, the 3.7 chip difference equals a 1100 meter difference in path delay. According to the GPS positions the balloon’s altitude increased by 619 meters (from 13205 to 13824 meters) during that time (see section 4.4.3 for more details).

### 4.4 GPS Bistatic Radar Altimeter

#### 4.4.1 Path Delays

A balloon is not a very steady platform, particularly not with the payload hanging several feet below (~50 ft.) swinging with the wind, and therefore satellites will drop in and out stressing the re-acquisition algorithms in the receiver. An interesting fact is that PRN13 has an elevation of 5 degrees and thus goes in and out as the balloon swings. By excluding PRN13 from the position solution the number satellites used for the position is constant and stable continuous 3D position solutions are acquired throughout the data set. Figure 14 shows the altitude from the point where enough satellites are available for a 3D solution.
4.4.2 Ground Tracks

The specular points for PRN5, PRN10 and PRN 30 as well as the ground track of the balloon is illustrated using Google Earth in figure 16. The yellow circles in the figure are spaced 5 miles apart and are centered around the start of the ground track from the balloon.

Figure 16: Ground tracks for three satellites and the balloon itself (from Google Earth)

4.4.3 Comparison of GPS Positions and Altimeter Data

Since the altimeter measures height above the specular points the GPS elevations have to be adjusted by removing the ground level. This is done by subtracting the GTOPO30 ground level for the balloon’s ground track from the balloon’s elevation. The NordNav receiver uses the ellipsoid as a reference, and GTOPO30 is based on geoid measurements, therefore the balloon elevation is also corrected for the difference between ellipsoid and geoid height. In the area where the data was collected the geoid is about 18.5 meters lower than the ellipsoid height. By using the corrected GPS position altitude a comparison between height above specular points and GPS height above ground can be made. However, since the specular points are located far away from the balloon’s ground track there will be a height difference here as well. The balloon and the specular point from each satellite move through five 1x1 km GTOPO30 tiles, and the elevation range for 5 tiles is ~25 m for PRN30 for example, as depicted in figure 17.

Figure 17: Ground level from GTOPO30

By comparing GTOPO30 ground levels from specular points and the balloon’s ground track, the altimeter heights can be corrected for this difference to give a more accurate reading of the actual altitude above ground instead of altitude above the specular point. Figure 18 below shows the altimeter values from three satellites as well as the corrected altitude from GPS positions for the entire time span (160 seconds).

Figure 18: Comparison of GPS positions and altimeter data
5 Conclusion and Results

5.1 Miniaturized Receiver

As with any design there are always trade-offs. For this design the trade-off when miniaturizing the receiver is mainly that the storage capacity becomes limited as well as the bandwidth and resolution. However, if a narrow spectral bandwidth is sufficient and if ~15-30 minutes of data is enough, then the miniaturized receiver is a very good alternative to other bigger and more expensive systems. For UAVs and other small platforms a miniaturized receiver might be the only option.

5.2 List of Major Components

- VIA EPIA NL Nano-ITX Single Board Computer [16]
- PicoPSU Miniature Power supply [18]
- SiGe Semiconductor PointCharger SE4110L GPS Front-End [17]
- Altera Cyclone EP1C12Q240 FPGA [19]
- Brain Technology USB HighSpeed Interface V2.6 [10]

5.3 GPS Bistatic Radar Altimeter

The altimeter has a bias of 26-47 m, depending on which satellite that is used. Assuming that the GPS positions are correct. A 20 second snapshot of the comparison between the altimeter and GPS positions is shown in figure 19, notice that the y scale is 20 meters.

It is not clear which error source is dominant, but it could be the error in specular point estimation. Because of the large distances and the signal scattering from the farm land the specular point might be misplaced causing an error that will propagate due to the geometry. GTOPO30 has a 1 km horizontal and 1 m. vertical resolution, which is less than ideal if the ground is not even. The balloon moved through 5 tiles in 4 minutes and the specular points are 10 to 15 miles away (16-24 km and the same number of GTOPO30 tiles).

Figure 20 depicts the direct and reflected waveform for PRN5 at an altitude of 25 km, this shows that the reflections can be used for altimetry at altitudes of at least 25 km with a relatively low integration time of 1000 ms (non-coherent integration).

Figure 20: Reflected waveform for PRN5, 25 km altitude

Acknowledgements

Thanks the people at EOSS for organizing the balloon flight (EOSS-103). We would also like to thank Staffan Backen and Tore Lindgren of the GNSS Research Group at Luleå University of Technology in Sweden for providing ground track plotting software compatible with Google Earth and scripts for specular point calculation.

References


